National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

University of Alberta

MACHINE LEARNING TECHNIQUES FOR THE CONTROL OF FES-ASSISTED LOCOMOTION AFTER SPINAL CORD INJURY

by

ALEKSANDAR KOSTOV

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy

DIVISION OF NEUROSCIENCE

Edmonton, Alberta

Fall 1995

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

THE AUTHOR HAS GRANTED AN IRREVOCABLE NON-EXCLUSIVE LICENCE ALLOWING THE NATIONAL LIBRARY OF CANADA TO REPRODUCE, LOAN, DISTRIBUTE OR SELL COPIES OF HIS/HER THESIS BY ANY MEANS AND IN ANY FORM OR FORMAT, MAKING THIS THESIS AVAILABLE TO INTERESTED PERSONS.

L'AUTEUR A ACCORDE UNE LICENCE IRREVOCABLE ET NON EXCLUSIVE PERMETTANT A LA BIBLIOTHEQUE NATIONALE DU CANADA DE REPRODUIRE, PRETER, DISTRIBUER OU VENDRE DES COPIES DE SA THESE DE QUELQUE MANIERE ET SOUS QUELQUE FORME QUE CE SOIT POUR METTRE DES EXEMPLAIRES DE CETTE THESE A LA DISPOSITION DES PERSONNE INTERESSEES.

THE AUTHOR RETAINS OWNERSHIP OF THE COPYRIGHT IN HIS/HER THESIS. NEITHER THE THESIS NOR SUBSTANTIAL EXTRACTS FROM IT MAY BE PRINTED OR OTHERWISE REPRODUCED WITHOUT HIS/HER PERMISSION.

L'AUTEUR CONSERVE LA PROPRIETE DU DROIT D'AUTEUR QUI PROTEGE SA THESE. NI LA THESE NI DES EXTRAITS SUBSTANTIELS DE CELLE-CI NE DOIVENT ETRE IMPRIMES OU AUTREMENT REPRODUITS SANS SON AUTORISATION.

ISBN 0-612-06571-5

Canada

University of Alberta
Edmonton
Canada T6G 2H1

Department of Computing Science
Faculty of Science

615 General Services Building, Telephone (403) 492-5198
FAX (403) 492-1071

October 4, 1995

Dr. Aleksandar Kostov
Faculty of Rehabilitation Medicine
3-48 Corbett Hall
University of Alberta
Edmonton, Alberta
T6G 2G4

Dear Aleks:

It is a pleasure to authorize you to use in your thesis, and anywhere else you need them, the Figures B2 - B6 in Appendix B of my paper with Jan Gecsei: Adaptation algorithms in binary tree networks, which appeared in IEEE Systems, Man and Cybernetics 1979.

Congratulations on finishing an excellent piece of work on your thesis!

Sincerely,

W. W. Armstrong

William W. Armstrong,
Professor

University of Alberta

Library Release Form

Name of Author: ALEKSANDAR KOSTOV

Title of Thesis: MACHINE LEARNING TECHNIQUES FOR THE CONTROL OF FES-ASSISTED LOCOMOTION AFTER SPINAL CORD INJURY

Degree: Doctor of Philosophy

Year this Degree Granted: 1995

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Kosta Aleksandar

#206 11104 84 Avenue

Edmonton, AB, T6G 2R4

Date Oct. 6/1995

To all those who may be helped to walk again.

"In rehabilitation, for every mile on the road there are two in the ditch."

-- Linda Walters

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled:

MACHINE LEARNING TECHNIQUES FOR THE CONTROL OF FES-ASSISTED LOCOMOTION AFTER SPINAL CORD INJURY

submitted by ALEKSANDAR KOSTOV in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. Richard B. Stein

Dr. Dejan B. Popovic

Dr. William W. Armstrong

Dr. Brian J. Andrews

Dr. Arthur Prochazka

Dr. Howard J. Chizeck

Dr. William F. Dryden

Date Oct. 4, 1995

Abstract

Functional electrical stimulation (FES) has been used as a substitution for the missing neural excitation of the paralyzed muscles after spinal cord injury. FES-assisted walking with preset stimulation patterns is usually controlled by a therapist or the subject using manual controls mounted on the handgrips of a walking aid. Automation of the switching control can be done by designing a rule-based control algorithm which will replace the decision making process the person uses to control the stimulation manually. These rules are usually designed by intuitive 'hand-crafting' and by applying them on a set of sensory feedback signals. This process has to be repeated for each subject due to highly specific disabilities resulting from physically similar injuries.

In this thesis a method is proposed, developed, and applied for automatic generation of control rules, which may provide a much faster evaluation of new subjects than the "hand-crafting" method. The rules are extracted from a set of sensory feedback signals and stimulation control signals recorded during FES-assisted walking controlled by a skilled therapist or the subject. The rule-generation method is evaluated using two different machine learning techniques, Adaptive Logic Networks (ALNs) and Inductive Learning (IL). Very fast training and high generalization of both techniques justified the design of the integrated control system (ICS). The ICS, currently based on ALNs, provides an efficient tool to acquire sensory and control signals, to process these signals, to train the ALNs in mapping the control function, to test the trained ALNs and to use them for control signal generation in real-time control of the FES-assisted walking of subjects with incomplete spinal cord injury. The IL technique was also evaluated in rule-generation for control of walking of subjects with complete spinal injury and its potential for cloning the subject's skill in switching the stimulation was demonstrated. In addition, ALNs were evaluated for continuous control of single joint flexion-extension, based on signals recorded from natural sources, such as nerves and muscles of cat's hind limb. Through experimental work it has been demonstrated that both techniques are able to generate control rules quickly and to generalize, not only over daily subsequent walking sessions but also over the sessions occurring several days after the training. This provides a good basis for design of robust control systems for FES-assisted walking.

Preface

The format of this thesis may be unusual due to the need to keep track of several different phases of the thesis project and at the same time to avoid repeating the introduction and the methods and materials section. The thesis addresses the FES-control problem; it has only one hypothesis and the approach to the resolution of the problem stays consistently the same. The only thing that changes throughout the thesis is the level of completeness of the problem's resolution. The thesis project started as a nice idea for an afternoon's test in late 1990, and gradually grew up from the size of small report at the local motor control meeting to the level of Student Research Award Winner at the International Federation for Automatic Control meeting in the beginning of 1994 and two finalist awards in Student Paper Competitions at the 14th and 17th Annual International Conferences of the IEEE Engineering in Medicine and Biology Society. The feasibility test and the evaluation of the whole approach to the FES-control problem were performed at the same time as the development of the system components. Therefore, different phases of project were conducted with components at different stages of development. In particular, this applies to the Adaptive Logic Networks (ALNs), which were evaluated for the first time in their original Atree 2.0 and Atree 2.7 version. As a result of these tests the original Atree 2.7 was modified resulting in superior performance. Then, close to finishing the integrated control system (ICS), a new ALN learning technology was introduced and released in the form of version Atree 3.0 beta 1.0. This in turn successfully passed the test and was incorporated into the final version of the integrated control system. For these reasons, the thesis is structured in a "Mixed Format", i.e. a combination of the Traditional and the Paper Format. The major difference between this format and the Traditional format is that each of the experiments, presented as a section of the Results chapter in the body of the thesis, has its own Methods and Materials section. Furthermore, each experiment refers to the main Methods and Materials chapter.

Acknowledgment

The support and help of colleagues is critical to a project of this scale. I wish to thank my supervisory committee: Richard B. Stein, Dejan Popovic, William W. Armstrong and Brian J. Andrews for their guidance, support and patience throughout. I would also like to thank Bill Armstrong for opening a new and wonderful world of neural networks to me, and for providing a chance to participate in an exciting development of Adaptive Logic Networks. Also, my thanks to Monroe Thomas for an excellent programming job and to Ben Heller for providing his program EMPIRIC.

My heartfelt thanks to Linda Walters, who participated in this project from the beginning up to the present time. I have always appreciated her words of encouragement.

Also, many thanks to the training committee of the Neuroscience Network of Centres of Excellence for their unreserved support.

And at the end I thank my family: Ibojka, Sanja and Jelena. Their enormous understanding and confidence in me as well as practical help during technical preparation of the manuscript has been a lasting gift providing me with the motivation to work hard and to finish this project. I hope I have fulfilled their expectations.

TABLE OF CONTENTS

1. INTRODUCTION **1**

1.1 THE ROLE OF HUMAN WALKING 1

1.2 WALKING AFTER SPINAL CORD INJURY 2

1.3 A BRIEF HISTORY OF ELECTRICAL STIMULATION 9

1.4 PHYSIOLOGY OF ELECTRICALLY STIMULATED MUSCLE 13

1.5 ELECTRICAL STIMULATION METHODOLOGY 16

1.5.1 Stimulation parameters *16*

1.5.2 Electrodes *20*

1.6 APPLICATION AND CONTROL OF FES IN LOCOMOTION 21

1.6.1 Control methods for a neural prosthesis *28*

1.6.2 The importance of system modeling *33*

1.6.3 Finite-state models and its derivatives *34*

1.6.4 The derivation of control rules for RBC *38*

1.6.5 Automatic generation of control rules for RBC *39*

1.6.6 Related work *41*

1.7 INITIAL THESIS HYPOTHESIS AND OBJECTIVES 43

1.7.1 The initial hypothesis tested in this thesis *43*

1.7.2 The proposed approach to modeling of the control system *43*

1.7.3 The specific reasearch questions *44*

1.7.4 The major practical objectives *44*

2. METHODS AND MATERIALS **46**

2.1 SUBJECTS 46

2.1.1 Selection of candidates for FES-aided locomotion *46*

2.1.2 Subject L.W.: Case History *48*

2.2 EXPERIMENTAL ANIMAL 51

2.3 SELECTION AND DESIGN OF SENSORS 52

2.3.1 Studying walking *53*

2.3.2 Force sensors *60*

2.3.3 Goniometers *65*

2.3.4 Inclinometers *67*

2.4 SIGNAL CONDITIONING HARDWARE 67

2.5 DATA ACQUISITION AND I/O SYSTEM 69

2.6 DIGITAL SIGNAL PROCESSING 69

2.7 MACHINE LEARNING TECHNIQUES 71

2.7.1 Adaptive logic networks *73*

2.7.2 Inductive learning *81*

2.8 SOFTWARE SUPPORT 84

2.8.1 Evaluation of ALN V.2 - program 'WALKON' *84*

2.8.2 ALN V.2 and ALN V.3 implementation in the real-time control: program 'FESCONT' *88*

2.8.3 Program 'EMPIRIC' *90*

3. RESULTS **91**

3.1 MANUAL CONTROL OF FES FOR LOCOMOTION 91

3.1.1 GAIT ANALYSIS FOR MANUAL CONTROL OF STIMULATION *96*

3.2 HAND-CRAFTED RULE-BASED CONTROL OF FES 100

3.2.1 Control Mode 1 ..... *104*

3.2.2 Control Mode 2 ..... *105*

3.2.3 Practical implementation of Control Modes 1 and 2 ..... *106*

3.2.4 Gait analysis for automatic control of the stimulation *109*

3.2.5 Conclusion *111*

3.3 AUTOMATIC GENERATION OF CONTROL RULES BY MACHINE LEARNING TECHNIQUES: EVALUATION OF ADAPTIVE LOGIC NETWORKS (ALNs) FOR SWITCHING CONTROL 112

3.3.1 Adaptive Logic Networks *114*

3.3.2 ALN application *116*

3.3.3 Results *117*

3.3.4 Discusion and conclusion *120*

3.4 EVALUATION OF IL FOR SWITCHING CONTROL OF FES 122

3.4.1 Methods *122*

3.4.2 Results *126*

3.4.3 Discusion and conclusion *131*

3.5 ALN - IL COMPARISON 132

3.5.1 Methods *132*

3.5.2 Results *135*

3.5.3 Discussion *140*

3.5.4 Conclusion *142*

3.6 REAL-TIME SWITCHING CONTROL OF FES BY ALN 144

3.6.1 Methods *145*

3.6.2 Results and discussion *153*

3.7 EVALUATION OF ALNs FOR CONTINUOUS CONTROL OF FES USING BIOLOGICAL FEEDBACK SIGNALS 175

3.7.1 USING BIOLOGICAL FEEDBACK SIGNALS FOR SWITCHING FES-CONTROL *175*

3.7.2 CONTINUOUS FES-CONTROL USING BIOLOGICAL SENSORY FEEDBACK SIGNALS *187*

4. DISCUSSION, CONCLUSIONS AND THE FUTURE WORK **201**

5. BIBLIOGRAPHY **208**

APPENDIX A: DATA ACQUISITION SYSTEMS **A-1**

APPENDIX B: THEORETICAL BASIS FOR MACHINE LEARNING TECHNIQUES B-3

APPENDIX C: PROGRAMS DEVELOPED FOR THIS THESIS PROJECT C-36

List of Tables

Table 1.2.1. Typical impairments and functional capacities after spinal cord injury (modified from Cull and Hardy, 1977). 7

Table 1.2.2. Classification of impairment after spinal cord injury (modified from Frankel et al. 1969). 8

Table 2.1.1. Information about the subjects participating in the study 47

Table 2.6.1. The fastest strides measured in data recorded from subjects participating in this study and the corresponding cut-off frequencies. 70

Table 2.7.1. Basic characteristics of different types of binary encoding. 80

Table 2.8.1. Main functions of the program FESCONT and their description. 89

Table 3.2.1. Statistical values of the timing of major events in a gait cycle related to the switching control of walking (number of steps N=17). 102

Table 3.2.2. Average gait cycle parameters for manual and automatic (Control Mode 1) control of the stimulation (Subject L.W.) 110

Table 3.3.1. ALN inputs and output. 116

Table 3.5.1. Average training and test errors for all ALN and IL learning sessions. 135

Table 3.5.2. A summary of the comparison between ALN and IL. 142

Table 3.7.1. VAF as a measure of ALN learning performance and generalization on test data. 198

List of Figures

Figure 1.2.1. The relationship of the spinal cord and the spinal nerves to the vertebral column. Injury at the upper pointing hand affects mostly the spinal cord and a few spinal nerves, while injury at the lower pointing hand can affect many spinal nerves in addition to the spinal canal (modified from Rieser et al., 1985). 4

Figure 1.2.2. Distribution of spinal cord injuries according to affected vertebrae. Injury above the pointing hand produces quadriplegia while injury below the line produces paraplegia (modified from Benes, 1965). 5

Figure 1.5.1. Stimulation parameters for unidirectional current or voltage stimulation. 17

Figure 1.6.1. Modular structure of the Neuroprosthetic Device (ND) and its interface with the user. 24

Figure 1.6.2. Various types of control systems: (A) Open-Loop Control (Reference-based); (B) Closed-Loop Control (Error-Driven); (C) Combined Reference-Based Closed-Loop Control (Error-Driven). The desired trajectory may be required for the design of the reference-based controller, but does not act as an input during controller operation; this relationship is indicated by the dashed lines in (A) and (C) (modified from Tashman and Zajac, 1992). 30

Figure 1.6.3. The model-based control system (modified from Tashman and Zajac, 1992). 31

Figure 2.3.1. Temporal events in a gait cycle (modified from Perry, 1992). 56

Figure 2.3.2. The construction of force sensing resistor (FSR). 61

Figure 2.3.3. Force sensor and its connections. 62

Figure 2.3.4. Shoe insole instrumented with force sensing resistors. 64

Figure 2.3.5. Twin axis goniometer M180. This goniometer measures the relative position of the two end blocks in the X and Y planes. 66

Figure 2.3.6. Twin axis goniometer M180 installed across: A) hip joint to measure its abduction-adduction and flexion-extension angles; B) knee joint to measure its flexion-extension angle; and C) ankle to measure its flexion-extension and inversion-eversion angle variations. 67

Figure 2.4.1. Schematics of the signal conditioning electronics for force sensors. A 50-pin connector provided mechanical coupling with the rest of hardware interface presented in Figure 3.6.2. 68

Figure 2.7.1. Data preparation for use with MLTs, and the sequence of MLT implementation phases in control applications. 72

Figure 2.7.2 Seven node ALN with three input variables. To enable synthesis of monotonic decreasing functions, complements of the input bits are also 74

present in the input vector (primed).

Figure 2.7.3. Binary representation of quantization levels. 78

Figure 2.7.4. Random walk encoding technique. 79

Figure 2.7.5. "Thermometer" or unary encoding. 81

Figure 2.7.6. Inductive learning technique produces simple and comprehensible decision trees. X1 and X2 are input attributes (whose possible pairs of values represent the domain of the function to be learned). The pairs are classified into two classes (representing codomain). 82

Figure 2.8.1. Data preparation and MLT functions built into program WALKON, which is developed for evaluation of ALN V.2 functions. 85

Figure 3.1.1. Gait analysis of a subject L.W. walking with (A) a single channel of stimulation applied to the skin over the common peroneal nerve, (B) two additional channels applied over the quadriceps and gluteal muscles and (C) a single channel stimulator surgically implanted over the common peroneal nerve. Averages (solid lines) and standard errors (dotted lines) have been computed for the hip, knee, and ankle joint angles for 5,5, and 9 gait cycles in (A), (B), and (C) respectively. Each gait cycle began at the time of foot contact. For all angles, flexion is displayed upward with respect to the straight (180°) or right angle position (90°) (modified from Stein et. al, 1993). 97

Figure 3.1.2. Ground reaction force measured under the ball of subject's right foot presented together with the switching signal recorded from the manual switch used to control FES during walking. The FSR used for measuring ground reaction force is highly nonlinear, so only a qualitative scale is shown. 98

Figure 3.1.3. Detection of the events during FES-assisted gait. The first two traces are combined signals recorded from switches installed under the heel and under the ball of the foot of right and left leg respectively. The third signal represents manual stimulation control. 99

Figure 3.2.1. Gait cycle event and timing detection using foot-switches. The first two traces are combined signals recorded from switches installed under the heel and under the ball of the foot of the right and left leg respectively. The third signal represents manual stimulation duration control. Signal levels for "toe ON", "toe & heel ON" and "toe & heel OFF" are marked as well as time intervals which are used to define relationships between the start and duration of the stimulation and the position of the feet. 102

Figure 3.2.2. Gait cycle event and timing detection using only two force sensing resistors installed under the front of the foot in both legs. The first two traces are samples of the signals recorded from force sensors positioned under the central metatarsal areas of both feet. Force increases upwards. The signal in the third trace indicates the time intervals when the subject pressed on the manual switch to deliver stimulation to her left leg. 103

Figure 3.2.3. State diagram and transitional conditions for Control Mode 1 (CM1) and Control Mode 2 (CM2). CM1 includes all states and transitions except these in the shaded boxes. CM2 includes all the states and conditions presented. ***RFS*** and ***LFS*** stand for Right and Left Front Sensor and ***ht*** and ***lt*** stand for the higher and lower thresholds respectively. 105

Figure 3.2.4. The block-diagram of the implementation of CM1 and CM2 in discrete electronics. 107

Figure 3.2.5. Ground reaction force measured under the ball of subject's right foot presented together with the switching signal generated by the Control Mode 1 logic used to control FES during walking (Subject L.W.). 110

Figure 3.3.1. The concept of using machine learning techniques in control of FES-assisted locomotion. 113

Figure 3.3.2. A structure of machine learning system based on adaptive logic networks (ALN version 2.x). 115

Figure 3.3.3. An example of the signals recorded from four force sensors and manual switch during FES-assisted walking controlled manually by the subject. 116

Figure 3.3.4. Adding more inputs can increase the number of correct predictions. 118

Figure 3.3.5. Adding past data can increase the number of correct responses during training, but may not during testing on new data. The number of samples was varied from only current one to five (current sample plus up to four previous samples at 0.4, 0.8, 1.2 and 1.6 s before the current sample). 119

Figure 3.3.6. The ALN can predict future patterns of stimulation on the data used for training, but the number of correct responses declines when tested on new data. 120

Figure 3.4.1. Signals recorded from six FSRs measuring ground reaction force (GRF), two biaxial inclinometers, each measuring two hip joint angles, and two manual switches during FES-assisted walking controlled manually by the subject A.M. 124

Figure 3.4.2. An example of IL prediction (dashed line) of actual stimulation intervals initiated by right hand switch closure (solid line) during FES-assisted walking. 126

Figure 3.4.3. Text form of IL decision tree showing IF(...) THEN(...) ELSE(...) rules and the extracted signal importance. 127

Figure 3.4.4. The average importance of sensors used in IL training. 128

Figure 3.4.5. Reduction of the number of sensors by inductive learning. 128

Figure 3.4.6. The use of previous signal samples in addition to the current ones significantly reduces the size of the decision tree and the generalization error. 130

Figure 3.4.7. The use of a differentiated signal and one previous sample in addition to the original four sensory signals resulted in the smallest decision trees and generalization error. 130

Figure 3.5.1. An example of training signals recorded in subject L.W. during FES-assisted walking. Sensors installed in subject's shoes (FS) measure ground reaction force in *N*, and goniometers (GM) attached across the joints of the stimulated leg measure joint angle variations in *deg*. Stimulation was controlled by the subject manually pressing on the switch (bottom trace). Multi-channel inputs are required for safe control but they make analysis of the recorded data difficult. 134

Figure 3.5.2. Optimization of ALN training with regard to training duration and test error. 136

Figure 3.5.3. Optimization of the IL decision tree with regard to its size (number of nodes) and test error. 136

Figure 3.5.4. The use of a single sample from the past in addition to the current one reduces the test error. The delay of the sample from the past should be optimized for a particular data set. It is recommended to use more than one sample from the past if computational power allows it. 137

Figure 3.5.5. It is possible to predict future stimulation events by both techniques. There is a simple rule in event prediction by the ALN: the longer the prediction time, the bigger the test error. Because of larger variability in IL results, the above rule does not always apply to IL. This example shows that a smaller error and a smaller variability was achieved by the ALN. 138

Figure 3.5.6. Adaptive logic in ALNs provides the user with a retraining capability. In this graph ALN retraining was compared to IL regular training and plotted as a reduction of the error resulting from retraining. The graph has 4 parts: 1) average training error for both ALN and IL techniques; 2) average test error for both ALN and IL techniques; 3) ALN test error after retraining; 4) performance error on the first part of the data file after retraining (compare with ALN result shown in part 1). 140

Figure 3.6.1. Integrated control system (ICS) for control of FES-assisted locomotion: A) ***Data acquisition***, ***ALN training*** and ***Off-line test***, B) Generalization of trained ALNs in real-time ***Walking test*** and ***Walking control***. 147

Figure 3.6.2. Electronic circuits interfacing stimulator, manual switch, sensors, signal conditioning devices, and PC-based ALN controller. 150

Figure 3.6.3. ALN training evaluation: A) ALN training evaluation with no functional errors; B) ALN training evaluation producing functional errors. 151

Figure 3.6.4. ***ALN training*** results applied on the training data set recorded in a control subject (A.K.). The subject was asked to simulate the stimulation signal required to activate flexion in his left leg. The simulated stimulation control signal (seventh trace) was produced by pressing on a manual switch. The two bottom traces are results of the evaluation of trained ALNs before and after restriction rules were applied. There are no functional errors (missing or extra stimuli) in the obtained output control signal (bottom trace). 154

Figure 3.6.5. ***Off-line test*** results on a new sequence of data recorded in a control subject (A.K.). There are no functional errors in the obtained output control signal (bottom trace). 155

Figure 3.6.6. ***Walking test*** results obtained by real-time evaluation of the trained ALNs during walking. A stimulation control signal predicted by the ALNs was presented to the subject in the form of a tone signal which sounded whenever the stimulator should be ON. As in previous steps, the subject was asked to simulate the stimulation signal required to activate flexion in his left leg. 156

Figure 3.6.7. ***ALN training*** results applied on the training data set recorded in an SCI subject (L.W.). The subject controlled the stimulation by pressing on a manual switch installed on a grip of her four-point wheeled walker. The stimulation control signal (seventh trace) was used to activate a flexion withdrawal reflex in her left leg. The two bottom traces are results of the evaluation of trained ALNs before and after restriction rules were applied. 159

Figure 3.6.8. ***Off-line test*** results on a new sequence of data recorded in an SCI subject (L.W.). Although there are some glitches at the end of several stimuli in the second trace from the bottom, there are no functional errors in the obtained output control signal (bottom trace) due to application of restriction rules. 160

Figure 3.6.9. ***Walking test*** results obtained by real-time evaluation of the trained ALNs during walking. The stimulation control signal predicted by the ALNs was presented to the subject in the form of a tone signal which sounded whenever the stimulator was supposed to be ON. As in previous steps, the subject was asked to manually produce the stimulation signal required to activate flexion in her left leg. 161

Figure 3.6.10. ***Walking control*** test performed after three steps that the subject controlled manually. Not being physically fit for walking, and having not walked because of an acute illness for several months, the subject was very tense which was reflected in the quality and number of steps taken. This was the reason behind having short walking sequences with long resting periods. After the walking session the subject admitted that she carried more than the usual part of her body weight on her upper extremities, which resulted in continuous turning of her walker to the right. That was another reason to stop after every few steps to correct the direction of walking and the walker. There were no functional errors in this walking session. 162

Figure 3.6.11. The second ***Walking control*** test demonstrates better subject's confidence in the control system, longer walking sequences and shorter resting intervals. Also, there was less load on the subject's arms, which resulted in less turning of the walker and a straighter walking path. After the sixth step (two manually and four ALN controlled), the subject accidentally, pressed on the manual switch. This caused instability and two extra stimuli (see arrow). Although a situation such as this one is potentially dangerous, we helped the subject to restore her balance and she continued walking. An additional restriction rule remains to be added which would prevent events such as this, but at the time of the experiment it did not exist. 163

Figure 3.6.12. ***ALN training*** results on signals recorded during manually controlled walking in one direction. No functional errors were generated by ALNs and all stimuli were approximated successfully. 164

Figure 3.6.13. ***Off-line test*** of the trained ALNs on the second half of the same walking round used for training. Although turning around was not part of the training data set, ALNs generalized well during turning around without stimulation and during walking back to start position. No functional errors or obvious dissimilarities between the original stimuli and ALN-produced ones are present. 165

Figure 3.6.14. ***Walking test*** results obtained by real-time evaluation of the trained ALNs during walking. The subject used stimulation not only for walking straight, but also during turning around for taking steps backwards. ALNs generalized well and without functional errors both during straight walking and during turning around.Figure 3.6.15. ***Walking control*** test performed after two steps controlled manually by the subject. This walking sequence was also very short due to the subject's fatigue. There were no functional errors in this walking session. 166

Figure 3.6.15. ***Walking control*** test performed after two steps controlled manually by the subject. This walking sequence was also very short due to the subject's fatigue. There were no functional errors in this walking session. 167

Figure 3.6.16. ***Walking test*** done using ALNs trained during the Second session (tree days before this experiment). This was the first walking test of the ALNs trained before the current walking session. After initial insecurity (one early and one extra stimulus during the first step), the rest of the stimulation was predicted correctly. 170

Figure 3.6.17. The second ***Walking test*** done right after the first one. Obviously, the subject demonstrated nice periodic walking during which all stimuli were correctly predicted. 171

Figure 3.6.18. ***Walking control*** done using ALNs trained three days before this experiment. Walking was very smooth and the subject demonstrated a full understanding of the importance of the appropriate weight transfer, as a way to voluntary inform the controller about her intentions. 172

Figure 3.6.19. The second ***Walking control*** round started with small instability resulting in one early and one extra stimulus (similar to the one shown in Figure 3.6.17). After the ALNs were put in charge of controlling the stimulation, the walking became as nice and smooth as one in the previous walking round. 173

Figure 3.6.20. The third ***Walking control*** round suffered the same problem at the beginning as the previous one, and it was much shorter due to development of fatigue. Otherwise, the steps recorded were regular and periodic. 174

Figure 3.7.1. The amplitudes (top) of compound action potentials were measured with triphasic cuff electrodes on three different nerves: sciatic, tibial and superficial peroneal in three legs (different symbols) of two chronic cats. The potentials were elicited by stimulating the sciatic nerve and recording 177

from the other two nerves which are branches of it and by stimulating the branches while recording from the sciatic nerve. The four values relative to the mean for each nerve over the entire recording period were averaged and plotted. The amplitudes of EMG contamination on the other nerves (bottom) were also averaged in the same way when the sciatic nerve was stimulated maximally. The EMG contamination could easily be distinguished because of its greater latency and slower time course.

Figure 3.7.2. (a) Power spectral density calculated from EMG recorded from medial gastrocnemius (MG) muscle, while a chronic cat walked on a treadmill. (b) Corresponding spectra are shown from a nerve cuff on the tibial nerve without filtering. (c) After high-pass filtering with a cutoff at 1000 Hz. Note that without filtering the peak spectral density of the EMG is approximately 100 times that of the neural signal (10 times the amplitude), but the ratio can be reversed using a fourth order high-pass filter. 178

Figure 3.7.3. Recordings from superficial peroneal (SP) and tibial (Tib) nerves as well as medial gastrocnemius (MG) and tibialis anterior (TA) muscles of a chronic cat. This 20 s recordings were selected to indicate the distinct rhythmic activity in peripheral nerves and ankle muscles in various behaviors. 182

Figure 3.7.4. Threshold levels were set (dashed lines) to determine instants when processed electroneurograms (ENG) recordings from tibial and superficial peroneal nerves (upper two traces) crossed above these preset levels. Threshold crossings activated transitions in a simple rule-base with four states (third trace), as explained further in the text. Entering state 2 turned on circuits for a fixed period of time (trace 5) that correspond on average to the duration of suprathreshold activity of MG EMG (trace 4). Similarly, entering state 4 turned on a circuit for a period (bottom trace) corresponding to the suprathreshold period of the TA EMG (trace 6). Note that the circuit operated reliably with no false positive or negatives. 183

Figure 3.7.5. When the rule-base was connected to a stimulator and stimuli were applied (middle trace), artifacts were generated on the processed tibial ENG (top trace), even with filtering, but this could be greatly reduced (bottom trace) by switching in a blanking circuit. 184

Figure 3.7.6. To match the activity pattern of an ankle muscle, based on the signals from the tibial and SP nerves, we used an adaptive logic networks (ALN). Neural signals (upper two traces in Figure 3.7.4) were presented to the ALN as the input, together with an output which consisted of pulses (second trace in this figure) representing periods when MG EMG exceeded a threshold value. The data on the left half of the figure were used in training the ALN to reconstruct the binary representation of the EMG signal. The reconstructed result is in the third trace, and the error is in the fourth trace. The right half of the figure presents test data, where reconstruction was not as successful as in training. Further details in the text. 185

Figure 3.7.7. Configuration setup for evaluation of ALNs for continuous control of FES. A) Training of the ALNs to approximate and predict target EMG signals from neural inputs during walking on a treadmill. B) Test of the trained ALNs on neural signals not used during the training. The ALNs are here predicting the muscular activity from indirectly related neural activity. 188

Figure 3.7.8. A sample of neural and muscular activity presented in the form of ENG and EMG signals recorded in the cat's hind limb during walking on the powered treadmill. A degree of shape irregularity is much higher in the EMGs than in ENGs. 189

Figure 3.7.9. Three-dimensional relationship between output MG EMG signal and sensory inputs (TI and SP ENGs) in approximately four gait cycles. 191

Figure 3.7.10. Three-dimensional relationship between output TA EMG signal and sensory inputs (TI and SP ENGs) in approximately four gait cycles. 192

Figure 3.7.11. Amplitude distribution for the MG EMG signal. 193

Figure 3.7.12. Amplitude distribution for the TA EMG signal. 193

Figure 3.7.13. Expert encoding compared to uniform unary encoding. 194

Figure 3.7.14. ALN training error 195

Figure 3.7.15. ALN test error 196

Figure 3.7.16. ALN training time and test error relationships with the number of encoding levels and the size of ALN trees. 196

Figure 3.7.17. An example of input and output signals used in ALN training and test. The colored areas in the last two traces show the difference between original and predicted EMGs during training and test. 199

Figure B.1. Structure of binary tree representing idealized classification function (from Armstrong and Gecsei, 1979, permission obtained from W.W. Armstrong) B-25

Figure B.2. Non-disjoint binary tree with inversions (from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong) B-26

Figure B.3. Disjoint balanced binary tree (from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong) B-26

Figure B.4. An example of two-level tree (from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong) B-26

Figure B.5. (A) Computer-controlled generation of assignments. Node functions g_k are stores in memory and communicated to the tree. (B) System of adaptive elements. Each element receives two function values from its successors, a heuristic responsibility signal s_k from its predecessor and desired tree output value $h(\xi)$ (last two only during the training). Assignments g_k are generated in each node from the above signals. (from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong) B-27

Figure B.6. Tree for threshold function $x \geq y$. (from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong) B-27

Figure B.7. Hierarchical decomposition of the input/output space into regions B-28

Figure B.8. Linear Threshold Element B-29

Figure B.9. A typical structure of logic gates and linear threshold elements B-30

Figure B.10. Linear threshold element can classify data presented in a form of a set of data samples or in form of a function. B-31

Figure B.11. Classification of data samples distributed in a non-linear way. B-32

Figure B.12. An ALN approximation of convex-type non-linear functions. B-33

Figure B.13. An ALN approximation of a complex function. B-34

Figure B.14. An ALN approximation of a complex function which does not have inverse function. B-35

Figure C.1. Opening screen of the program WALKON. C-44

Figure C.2. Opening a new data file. C-45

Figure C.3. Display single channel. C-45

Figure C.4. Stack all channels. C-46

Figure C.5. Set history points dialog box. C-46

Figure C.6. History points display using *Stack* function. C-47

Figure C.7. Truncate channels dialog box. C-47

Figure C.8. Random Walk encoding. C-48

Figure C.9. Unary (linear) encoding. C-48

Figure C.10. Arbitrary position of quantization levels over the amplitude range. C-49

Figure C.11. Dialog box for choosing a channel to shift. Similar dialog box is used for choosing a channel to differentiate. C-49

Figure C.12. Dialog window for specification of ALN training parameters. C-50

Figure C.13. Dialog window for specification of ALN tree parameters. C-50

Figure C.14. ALN training status window after the training was finished. C-51

Figure C.15. Evaluation of the trained ALN trees results in new data file spreadsheet. C-51

Figure C.16. Original stimulation control signal, the signal predicted by ALN and signal of their difference are organized in a new data file. C-52

Figure C.17. Assessing the performance of trained ALNs. C-52

Glossary

adaptive Boolean logic element: A node; an element of the ALN tree which can be any of four Boolean functions (AND, OR, LEFT, RIGHT).

adaptive logic networks: A type of artificial neural network for supervised learning which produces a binary decision tree.

cadence: Number of steps per minute during uniform speed walking.

cerebrovascular accident: Brain damage due to blood clot or hemorrhage; also known as 'stroke'.

class: Classification category; a value in the codomain.

codomain: The set of all possible values of output variables in supervised learning techniques.

decision tree: Training result of machine learning technique. It can be described by a 'rule-set'.

domain: The set of all possible values of input variables (attributes) to the machine learning technique.

epoch: A single presentation of the whole training input and output data set to the machine learning technique.

generalization: Performance of the learned function on new data unused during the training.

hemiplegia: Complete paralysis affecting one side of the body, usually due to brain disease or injury.

inductive learning: A supervised learning technique producing decision trees in the form of *IF(condition) THEN(a) ELSE(b)* rules.

input vector: Binary array in ALN consisting of all bits of input variables and their complements.

lazy evaluation ('parsimony'): One way of reducing the necessary evaluations to a small fraction of the elements in the network. Parsimony uses the simple fact that if one input of an AND gate is 0, then the output is also 0, and it is unnecessary to

evaluate the subtree producing the other input, nor indeed, even know its inputs. The same holds true for an OR gate and a 1 input, of course.

leaves: Nodes of the first layer of an ALN binary tree used to produce a binary input vector to the tree.

myoelelectric signal: Total signal representing muscular activity, seen at an electrode or differentially between electrodes; this signal is also called the electromyogram (EMG).

orthosis: An external mechanical support for some part of the body; also known as a 'brace'.

output vector: Binary array in an ALN consisting of output bits from all ALN trees.

paraplegia: Paralysis of the legs, usually due to disease or injury of the spinal cord.

paresis: Paralysis of muscles.

quadriplegia: Paralysis of both arms and both legs; also known as 'tetraplegia'.

rule: Conditional *IF(condition) THEN(a) ELSE(b)* structure where *a* and *b* represent *classes* (actions) or references to further *rules* of the same form.

rule-based controller: A controller in which the knowledge required to control the plant is as rules.

spasticity: Increased muscle tone due to involuntary contractions of muscles. Resistance of muscles to being stretched.

stroke: See 'cerebrovascular accident'.

supervised learning: A type of machine learning in which the learning algorithm is trained on sets of examples containing input and corresponding desired output signals.

test: Performance evaluation of the learned function on a test data set (new data not used during the training).

test error: A measure of generalization of the learned function expressed as a percentage of incorrectly predicted samples in the test data set or often as least-square error between desired and actual performance.

tetraplegia: See 'quadriplegia'.

training: Process of establishing an input-output mapping based on the set of inputs and their corresponding outputs.

training error: A measure of training performance expressed as a percentage of incorrectly classified samples of the training data set, or as least-square error.

voter: ALN V.2 has a feature to allow use of more than one tree for each bit in an output vector. The actual output bit is decided by majority vote of all trees producing a value for that particular bit.

Abbreviations

ABLE: adaptive Boolean logic element

ADL: active daily living

ALN: adaptive logic network

BTF: binary tree function

DBTF: disjunct binary tree function

DSP: digital signal processing

EMG: electromyogram (see ***myoelectric signal***)

FES: functional electrical stimulation

FSR: force sensing resistor

HAS: hybrid assistive systems

ICS: integrated control system

IL: inductive learning

LTE: linear threshold element

MG: medial gastrocnemius muscle

MLT: machine learning technique

ND: neuroprosthetic device

RBC: rule-based control

SCI: spinal cord injury or spinal cord injured

SP: superficial peroneal nerve

TA: tibialis anterior muscle

TI: tibial nerve

VAF: variance accounted for (%)

1. INTRODUCTION

1.1 THE ROLE OF HUMAN WALKING

Walking as a natural way of overground transportation from one place to another is one of the most complex and most important movements that humans do in their life. It is one of the most difficult movement tasks that we learn, but once learnt, it becomes almost subconscious. Also, it is obviously very difficult to approximate bipedal human-like walking by artificial walking machines, otherwise such machines would be used as ground transportation vehicles. What makes human walking advantageous over, for example, wheeled transportation systems is the fact that progression is possible over uneven terrain (i.e. obstacles, gaps in the surface, narrow edges). In practical terms, these advantages translate into easier access to buildings, confined spaces, etc. The task of copying the mechanism of human walking is not only difficult due to the complexity of the biomechanics of the musculoskeletal system, but also to the presence of the biological neural control system that takes care of the musculoskeletal system as well as the rest of the body to ensure reaching desired destination safely. Only when this complex neuro-musculo-skeletal system is disturbed by traumatic injury, neurological damage, gradual degeneration, or fatigue do we realize how limited is our understanding of the complex biomechanics and motor control mechanisms.

Inspired by the importance and the complexity of walking, David A. Winter wrote a beautiful poem in his book (Winter, 1991):

> "...Walking is for enjoying from one day to another - to play hopscotch, to play jump the rope, to play cops and robbers, to go dancing through the trees in the park, at parties for weddings and birthdays, with the beloved at nightfall, to meet the returning child..."

Jacquelin Perry, another prominent researcher in the field of gait analysis, wrote in her book (Perry, 1992) about the importance of walking and the difference between normal and impaired walking:

> "Walking is the body's natural means of moving from one location to another. It also is the most convenient means of traveling short distances. Functional versatility allows the lower limb to readily accommodate stairs, doorways, changing surfaces, and obstacles in the path of progression. Efficiency in these endeavors depends on free joints mobility and muscle activity that is selective in timing and intensity. Energy conservation is optimal in the normal pattern of limb action. Because of the numerous advantages of walking, patients strive to retain this capability even in the presence of severe impairment. As the various types of pathology alter mobility and muscular effectiveness, the patients substitute wherever possible, yield when they must, and accept compensatory reactions of adjacent segments as they occur. The resulting walking pattern is a mixture of normal and abnormal motions that differ in significance. Energy costs are increased and functional versatility is compromised."

1.2 WALKING AFTER SPINAL CORD INJURY

Spinal paralysis is one of the most devastating of all the illnesses that can befall man (Guttmann, 1976; Jackson, 1981; Rieser et al., 1985). Not long ago, 80% of subjects with spinal cord injury or disease died within three years due to subsequent medical complications (Jackson, 1981). Today, with modern treatment methods at spinal injury centers, persons with spinal cord injury can expect a normal life span. The incidence of persons with spinal cord injury requiring care is in the range of 8,000-9,000 new cases per year in the United States (approximately ten times less in Canada). With the increase of longevity post-injury, the prevalence of this problem is generally stimated to be in the range of 200,000 persons (Ozer, 1988). A typical subject with the spinal cord injury is a young male, under 30 years of age, injured in a motor vehicle accident (Oliver et al., 1988). The

second largest cause of the injury are amateur sporting activities, and diving is responsible for more than a half of such injuries.

A basic knowledge of the functional anatomy of the spinal cord is a prerequisite and a practical necessity for understanding spinal cord injury and for prescribing a proper clinical care. The spinal cord is a cylindrical extension of the central nervous system. The total length of the cord is about 45 cm in men and from 41 to 43 cm in women, while the length of the vertebral column is about 70 cm (Cayaffa, 1981). The spinal cord occupies approximately the upper two thirds of the vertebral canal. Thirty-one pairs of spinal nerves originate in the spinal cord. The spinal cord has been classically divided into five segments: cervical, thoracic, lumbar, sacral and coccygeal. Spinal cord segments, spinal nerves and vertebral body levels are traditionally marked using the first letter of the level's name and the level itself, for example, C5 equals to the fifth cervical level. Injuries and impairments can be classified according to either vertebral body levels or spinal cord segments. The spinal cord and the spinal nerves are illustrated in Figure 1.2.1. The main functions associated with spinal nerves are also marked.

The spinal cord has long been identified as an extremely trauma-sensitive organ. Seemingly moderate traumatic injury or tumor pressure is clinically associated with total and permanent paralysis (Osterholm, 1978). The spinal cord can be damaged either by injury to the spinal column, or this may happen without any noticeable changes in the bones of the spine. Most frequently, spinal cord injuries occur together with an injury to the spinal column. The spinal column is usually injured by one of the following three mechanisms (Benes, 1965): hyperflexion, hyperextension, or a direct blow to the spinal column. Both hyperflexion and hyperextension may combine torsion and lateral displacement. Figure 1.2.2 illustrates the distribution of injuries according to affected vertebrae. It clearly confirms the well established fact that injuries at the level of the fifth cervical and of the first lumbar vertebrae are the most frequent ones (Benes, 1965).

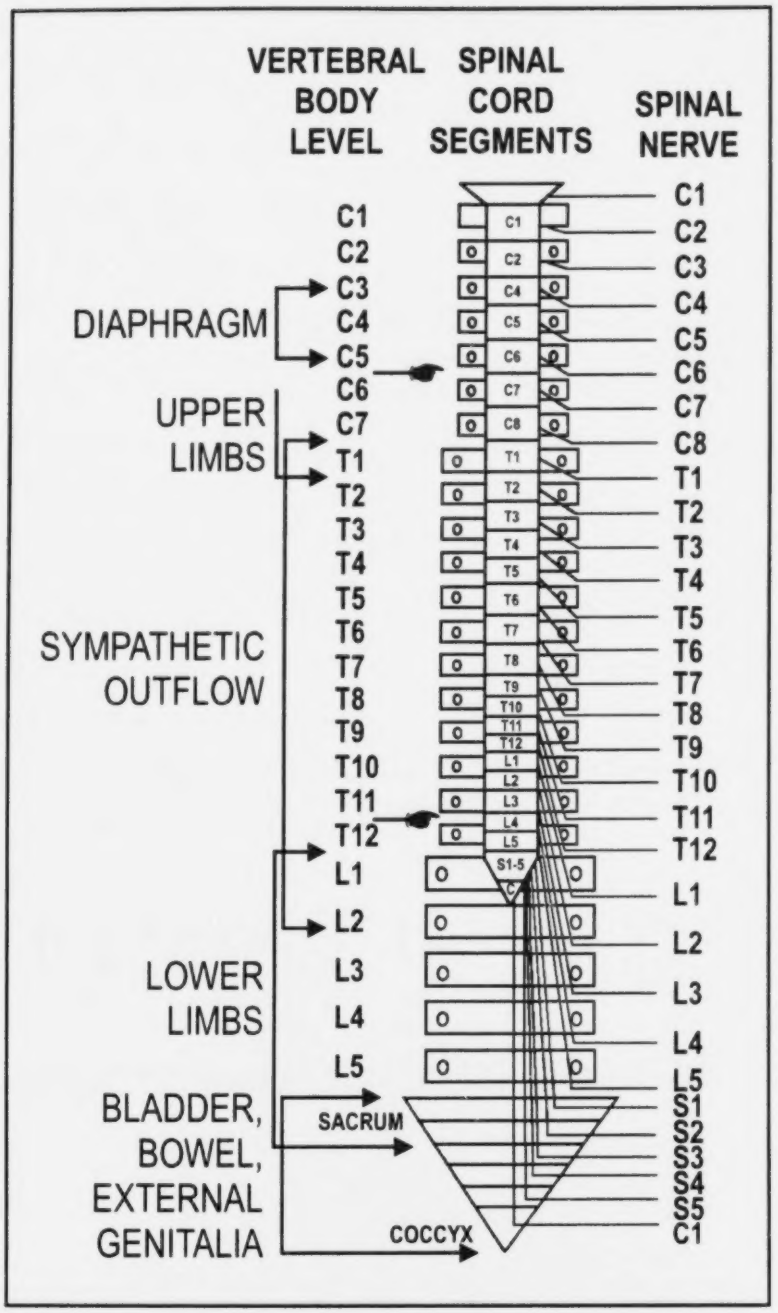

Figure 1.2.1. The relationship of the spinal cord and the spinal nerves to the vertebral column. Injury at the upper pointing hand affects mostly the spinal cord and a few spinal nerves, while injury at the lower pointing hand can affect many spinal nerves in addition to the spinal canal (modified from Rieser et al., 1985).

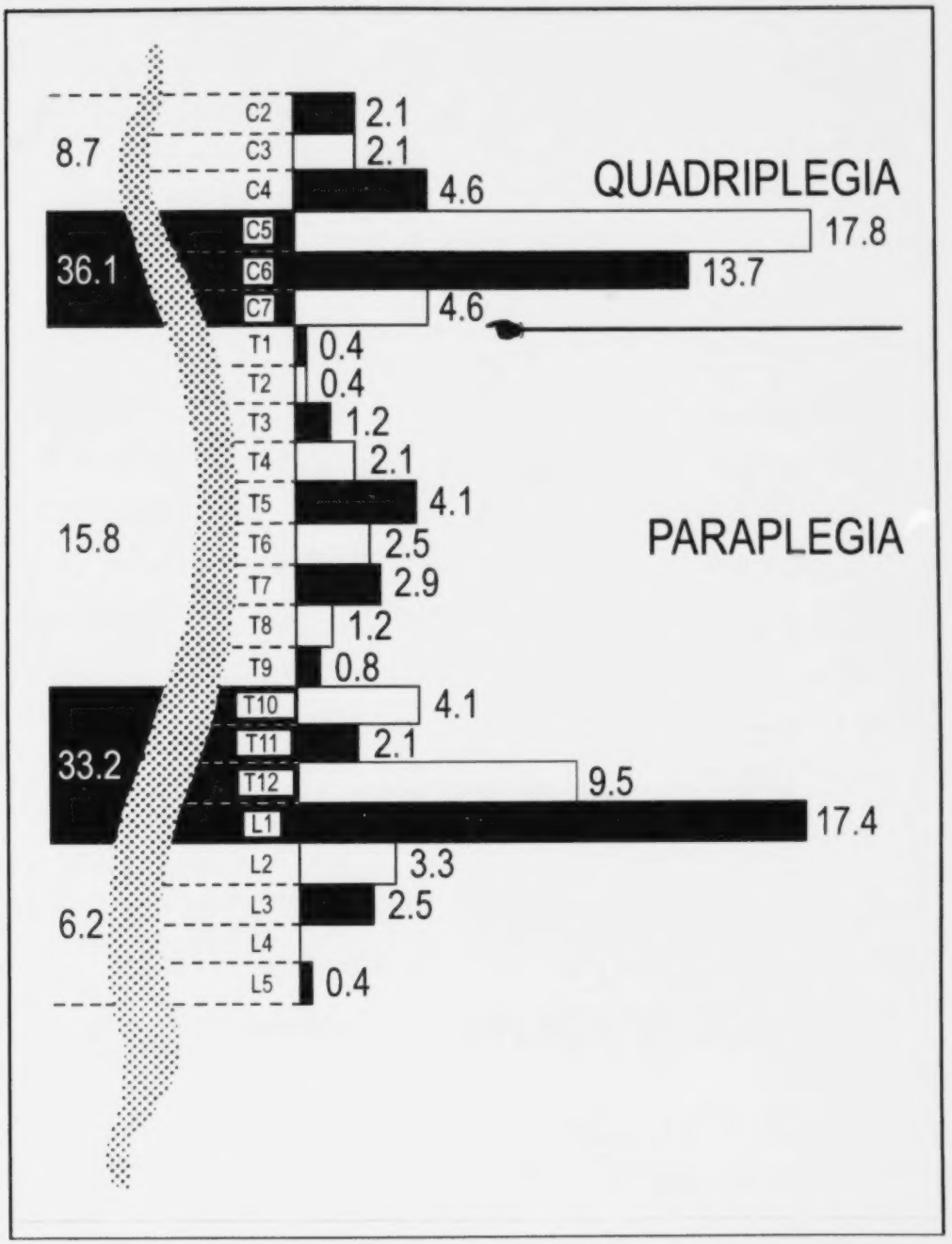

Figure 1.2.2. Distribution of spinal cord injuries according to affected vertebrae. Injury above the pointing hand produces quadriplegia while injury below the line produces paraplegia (modified from Benes, 1965).

In clarifying the interaction of the person and the injury, it is useful to use the well-established distinction made between ***'impairment'*** and ***'disability'*** (Ozer, 1988). 'Impairment' describes the loss of sensation or coordination of motor actions or loss of autonomic control of urination, bowel evacuation, and sexual function that may occur with damage to the spinal cord or roots, while 'disability' refers to the functional consequences of such impairments: the activities in the life of the person that have been affected. For example, the impairment in motor coordination called 'paraplegia' generally leads to a disability in mobility. It is important to recognize that it is frequently possible to effect change in these disabilities even if the impairments continue to exist.

Impairments resulting from injury are classified depending on which extremities are affected into: ***'quadriplegia'*** representing partial or total loss of sensation and/or motor functions in upper and lower extremities, and ***'paraplegia'*** representing the loss only in lower extremities (see Table 1.2.1).

Spinal cord injuries are usually classified as ***'complete'***, with a total loss of sensory or motor function below the level of the lesion; or ***'incomplete'***, with some preserved sensation or motor function below the level of the lesion. More precise and generally accepted classification of the injuries by the rehabilitation professionals is one proposed by Frankel et al. (1969), which defines five different grades of impairment as shown in Table 1.2.2.

The number of incomplete spinal cord injuries has been increased recently with the reduction in number of complete cases (Kralj and Bajd, 1989). This has been attributed to:

- improvements in emergency procedures to protect the spinal cord from further damage;
- seat belt legislation, which resulted in a reduction in death rate from motor vehicle injuries;
- improved motor vehicle engineering, which decreased the number of disastrous traffic accidents;
- preventive education of the people, before getting their drivers license, to avoid any rotary movement at the cervicodorsal and lumbodorsal junction when giving first aid to subjects with possible fractures of the vertebral column.

Table 1.2.1. Typical impairments and functional capacities after spinal cord injury (modified from Cull and Hardy, 1977).

IMPAIRMENT	FUNCTIONAL DESCRIPTION
C5 Quadriplegia	- totally dependent for many activities of daily living (ADL) - wheelchair necessary - reduced respiration function - elbow flexion possible - no finger movement - low endurance - lifetime attendant care necessary
C7 Quadriplegia	- assistance needed for ADL - wheelchair-bound - need splints and aids for wrist and forearms - has weak grasp - able to transfer with assistance
C8, T1 Paraplegia	- minimal assistance for ADL - wheelchair needed - may walk with 'swing to' gait with bracing - possible to drive adapted car
T7 Paraplegia	- independent in wheelchair - tight grasp - ambulation usually not functional - may drive a hand-controlled car - public transportation difficult
T12 Paraplegia	- completely independent - drives a hand-controlled car - normal walking gait not possible - uses hip-hiking with bracing - can climb stairs
L4 Paraplegia	- completely independent in self-care and ambulation - uses wheelchair by choice or for convenience - uses crutches for 'normal-type' walking - drives a hand-controlled car

Although this section deals only with spinal cord injury, there are many more diseases that produce similar effects to the spinal cord injury. Jacqueline Perry wrote on pathologies that affect walking (1992).

> "When poliomyelitis and amputations in otherwise healthy, young persons were the primary causes of gait abnormalities, it was sufficient to memorize few key action patterns. Now the clinical concerns relate to a far broader scope of pathology. Stroke, spinal cord injury, brain trauma, cerebral palsy, myelodysplasia, muscular dystrophy, geriatric amputation, degenerative joint disease, rheumatoid arthritis, multiple sclerosis, and complex patterns of mixed trauma comprise a representative but not exhaustive list."

It can be seen from Table 1.2.1 that serious injury at any of the levels in the spinal cord affects walking and mobility of the subject. The wheelchair provides a convenient low energy means of transportation in an environment specially designed or adapted for wheelchairs. However, the wheelchair only partially resolves the transportation problem. There are many more issues that a wheelchair-bound person has to deal with besides transportation: muscle atrophy, osteoporosis, pressure sores, bladder and kidney infections, etc. Assisted standing and walking, even in the form of physical exercise can significantly reduce these problems (Kralj and Bajd, 1989). Assisted walking, even if it may not be faster and more energy-efficient than a wheelchair transportation, can provide more independence at home or in places that are not specially adapted for people in wheelchairs. One way to provide such assistance to disabled people and to improve the quality of their lives is functional electrical stimulation (FES).

Table 1.2.2. Classification of impairment after spinal cord injury (from Frankel et al. 1969).

IMPAIRMENT	DESCRIPTION
Complete (A)	Complete motor and sensory lesion below marked segmental level.
Sensory only (B)	Motor paralysis is complete and some sensation is present below marked segmental level.
Motor Useless (C)	Some motor power is present below marked segmental level, but it is of no practical use to the subject.
Motor Useful (D)	Useful motor power is present below marked segmental level. Subjects in this group could move lower limbs, and many could walk, with or without aids.
Recovery (E)	Subject is free of neurological symptoms, i.e. no weakness, no sensory loss, no sphincter disturbance. Abnormal reflexes may have been present.

1.3 A BRIEF HISTORY OF ELECTRICAL STIMULATION

Treating disease with electricity has intrigued man for centuries. Early civilizations prescribed shocks naturally occuring from electric fish or rubbed amber, to cure ills ranging from headache to hemorrhage. Although the terminology of electrical stimulation may be a bit confusing (Kahn 1994), today's wide variety of stimulation apparatus all have in common one of two purposes: the stimulation of tissues for therapeutic or diagnostic purposes. Belonging to the first group are: stimulation of the muscles to contract or relax; stimulation of the nerves to produce analgesia or to increase dormant motion; and stimulation of the bone to enhance growth. Improved blood circulation is a common benefit from the stimulation of all these tissues. Stimulation for diagnostic purposes is used in neurophysiological tests such as measuring the conduction velocity of peripheral nerves, somatosensory evoked responses, etc.

Torpedo fish, capable of generating substantial electric shocks of 100-150 V, were recommended as therapeutic agents as early as 400 B.C. (Benton, et al., 1981). Topically applied to the head, these electric fish were reported to relieve headache. Placed under the feet, the torpedo cured arthritis. A diet including boiled torpedo was prescribed for asthma. In 46 A.D. Scribonius Largus employed the electrical discharges of the torpedo fish to treat gout and headaches, although it is not clear whether the results depend on stimulation or intimidation.

Amber, a fossilized resin, was noted by the ancients to produce shocks after being rubbed, and it was given as pills to cure inflammation, hemorrhage and nausea. A derivative of ***electrica***, from the Latin word for amber, became used in 1600 to describe the force which tingled the senses and moved the limbs. The first report on purposeful muscle contraction produced by static electricity was made in 1745 by Kratzenstein, a 21-year-old German physics graduate. He has applied static electricity to the paralyzed small finger of a woman which straightened out in 15 minutes.

In 1745 the invention of the Leyden jar, forerunner of the electrical capacitor, extended the applications of electricity by providing the ability to store energy and to conserve quantities of charge for later use. Various disorders, such as kidney stones, sciatica, epilepsy, and angina pectoris, as well as paralysis, were soon reported to be 'successfully' treated. A 1753 report by the practitioner Samuel Quelmalz describes a representative cure of the electric shocks:

> "A young man of 18 with hemiplegia of two years duration, was unable to stand or walk and had lost his speech. His fingers were held in involuntary flexion so that he was unable to put on his shoes by himself. His arm was motionless and his hand cold... I applied some shocks to his hand in the morning and again in the afternoon. After a few days he returned and was able to move the arm more freely and also to speak with greater ease. Electric shocks were given once or twice a week. Soon he recovered so much function that he no longer complained of inability to finger the violin as he had previously."

In spite of convincing testimonials of the therapeutic value of 'electric shocks', it was not until late 1700s that published reports linked muscle contraction to electrical stimulation of nerve. The first record of the use of electrical current to activate isolated muscles dates to 1791. Luigi Galvani (1737-1798) accidentally discovered that the leg of a frog twitched if it was touched with an electrically charged scalpel. He used electricity to activate an isolated nerve-muscle preparation. He also observed that application of dissimilar metals to the nerve of a frog muscle induced muscular contraction. Galvani assumed that the 'animal electricity' was generated by the nervous tissue and was stored in muscles, which he compared to Leyden jars, and that the metal only provided a path to discharge the inherent energy.

Subsequently, 'galvanism' or 'galvanization' were the names applied to the application of electrical current to the body. Galvani's nephew, Aldini demonstrated the use of direct-current stimulation for twitching the muscles in the decapitated head of an executed criminal. This event linked electricity and life and it was believed that electricity could restore life, a process called

'reanimation'. Soon, all kinds of excitable tissues became the focus of attention and there arose the need for controllable electrical stimuli to induce activity.

The first electrical stimulator was the capacitor or static electric machine. The most successful of these was the rotating-disk type due to Ramsden (c. 1768) which was used by Galvani in experiments mentioned above. The combination of a static-electricity machine and the capacitor (Leyden jar), which could store the charge produced by a static-electricity machine provided the opportunity of delivering single stimuli of controlled intensity and duration which enabled discovery of the strength - duration relationship of the stimulation.

Then the electromechanical cell connected to a switch to initiate and arrest current flow was invented. Special switches, called rheotomes, were developed to control the repetition and duration of current flow. However, it was the discovery of magnetic induction by Faraday in 1832 that paved the way for creation of the most controllable stimulator for the time being, the inductorium.

In 1795 Humboldt proved on his own muscles that their contraction resulted from direct stimulation. He also indicated that the nerve must be intact to achieve such a response. In 1799 Volta invented a dependable source of continuous electric current - 'Volta's pile'. With his 'pile' he noted that "a contraction takes place only at a first flow of electricity, and sometimes also at the breaking of the circuit..." This finding was tested and confirmed by Ritter, who concluded that if the exciting stimulus is not applied with briskness, the muscle would not contract. Combination of electrical stimulation and ancient Japanese practice of acupuncture produced a hybrid technique - electropuncture, proposed by Sarlandiere in 1825 and mastered by Duchenne de Boulogne. The latter became so interested in the technique that he devoted much of the remainder of his long life to electrical stimulation. He soon found that he could stimulate muscles electrically without piercing the skin and devised cloth-covered electrodes for surface stimulation, the basic design of which is still used. He was the first to use the alternating current for stimulation and he suggested

the word 'Faradic' to describe that current. Although noticing presence of "certain spots along the surface of the body and limbs that give peculiar responses to the electrode in producing ample muscle contraction", he did not offer any written explanation of this findings. It was left to Remak to explain that these famous points of election on the muscles are the points of entry of the muscular nerves. Mapping of the whole surface of the body by marking the motor points on the skin of subjects with silver nitrate was done by Von Ziemssen in 1857. By dissection immediately after death, he confirmed the correspondence of his clinical markings with the actual entrance of nerves into muscles.

During the 19th century many other investigators explored electrical stimulation for diagnosis, discovering that diseased muscles responded differently than normal muscles. The early 20th century is remembered as a time when a number of electrical stimulating devices were developed and produced commercially, mainly in Europe for electrodiagnosis and physical therapy.

The use of electrical stimulation as an aid in overcoming paralysis in human legs has intrigued many investigators and inventors which resulted in patents such as one filed for in 1951 by Giamo and entitled "Electrical Control of Partially Denervated Muscles", which illustrated surface electrodes for stimulation of leg muscles. In 1956 Browner filed for an "Ambulatory Electrical Muscle Stimulating Device", which incorporated the stimulator directly on a surface electrode assembly.

Although Keegan Jr. filed in 1961 on a "Device for Producing Muscle Therapy" which used surface electrodes over the pretibial muscles and a heel switch for control of the stimulator for footdrop, the tribute for the first effort to apply electrical stimulation as an aid to recover function in a disabled person is granted to Liberson (Liberson et al.,1961). In 1965 Offner and Liberson applied for a patent entitled "Methods of Muscular Stimulation in Human Beings to Aid in Walking" to cover the equipment they designed, which is very similar to Keegan's design. These early prototypes led to the development of a commercial system called 'Theratron' which was not very successful

because of problems with electrodes and excessive pain associated with the stimulation. Around 1962 a group at Case Western Reserve University (Reswick, Vodovnik, Crochetiere, Long and others) began to explore the use of electrical stimulation to obtain useful function in paralyzed subjects. Successful work of this group resulted in dissemination of the idea of using electrical stimulation to substitute for missing voluntary excitation and control of paralyzed muscles around the world.

Regarding stimulation-assisted locomotion, the flexor withdrawal reflex was known for a long time before it was suggested by Kabat (1954) to utilize it in facilitation of voluntary movements in spastic paralysis. The first electrical induction of the flexor withdrawal reflex during the swing phase in locomotion of hemiplegic subjects was tested by Lee and Johnston (1976). They have tested different stimulation sites and stimulation parameters and determined the best site to apply the electrical stimulus as well as the optimal combination of stimulus parameters to control the magnitude of the reflex. It is interesting that they found the sole of the foot to be the site which produced the longest duration of flexion response after stimulation.

More technical details about the development of the first stimulators can be found in an excellent review by Geddes (1994) and more clinical information on the early use of electric stimulators is provided in reviews by Reswick (1973), Benton et al. (1981) and Hambrecht (1992). Ever since these first steps of electrical stimulation implementation for assistance in functional movement, the field of functional electrical stimulation is constantly expanding and there are already many commercially available devices which originated in those ones mentioned above.

1.4 PHYSIOLOGY OF ELECTRICALLY STIMULATED MUSCLE

The decision to perform voluntary movement originates in the brain. The command is then sent in form of nerve impulses (action potentials) over the spinal cord, peripheral nerves, and neuromuscular junction to a skeletal muscle. Cell bodies of peripheral nerves, or *alpha motor*

neurons, are found in the various sections of the spinal cord. The impulses are conducted between the brain, either directly to these alpha motor neurons or onto a second set of neurons called *interneurons*. The spinal cord contains millions of interneurons. A theoretical description of the propagated action potential was given by the model proposed by Hodgkin and Huxley (1945, 1952). From the neuromuscular junction the signal spreads in all directions throughout the muscle cell membrane and through the transverse tubular system into the inside of muscle cells. Calcium is then released, and cross-bridges between the actin and myosin filaments are formed. The final task of the muscle is to generate force. According to current concepts of muscle physiology (Rothwell, 1987), the amount of force developed within a muscle is determined by the number of cross-bridges formed between the filaments as they slide past each other during a contraction. The resulting force depends not only on the command at the neuromuscular junction, but also on the muscle length, the velocity of shortening and the state of activation of the muscle before receiving the command.

The generation of force is also the aim when paralyzed skeletal muscles are stimulated electrically. In case of spinal cord injury the normal pathways in the spinal cord are damaged. One way to activate paralyzed muscle is by bypassing the neural lesion electronically. The main decision to make a movement is still made in the subject's brain, but the way the command is given and the path of the command before it reaches the muscle is different. The command which starts electrical stimulation can be given using hand or foot switches. Electrical stimuli are generated in the stimulator and delivered to the peripheral nerve through surface or implanted electrodes. From here on, the command path is the same as described for voluntary contraction of skeletal muscle, although not necessarily resulting in stimulation of the same muscles.

The human muscle is usually a combination of slow fibers capable of sustaining low levels of contractile activity without fatigue for prolonged periods and fast fibers capable of developing large forces, but fatiguing so rapidly that they can be used only in intermittent activities. Often in the literature, ***slow-twitch*** or ***type I*** fibers are denoted as ***red***, while ***fast-twitch*** or ***type II*** fibers are

called ***white*** fibers. Red fibers contain myoglobin, which gives them a characteristic color and serves as a reserve of oxygen. The duration of the twitch contraction is prolonged in slow-twitch fibers. They are also smaller in cross-sectional area, have a high capacity for oxidative metabolism, and a low capacity for glycolytic metabolism. In contrast, fast-twitch glycolytic fibers are large in cross-section, they have a low capacity for oxidative metabolism, and a high capacity for glycolytic metabolism. Fast-twitch fibers exhibit a twitch contraction of short duration. Some muscles are predominantly made up of white fibers, some predominantly of red, and some of a given mixture of the two.

All muscle fibers innervated by the same motoneuron have been found to be of the same histochemical type. Motoneurons innervating predominantly slow muscles discharge at a low frequency (10 to 20 Hz), and those supplying fast muscles at a higher frequency (up to 60 Hz). Motoneurons activating slow types of muscle are in general called **tonic**, while **phasic** types of motoneurons act on fast muscles. Tonic motoneurons have axons of small diameter, while phasic motoneurons have axons with a larger diameter.

In a voluntary contraction of normally innervated muscle, the slow-twitch fibers are recruited first, and only if increased tension is required, the fast-twitch fibers are then recruited. Slow muscle fibers are, therefore, activated frequently, while fast fibers are employed only during a burst of intense activity. When applying electrical stimulation, fibers with a greater diameter respond earlier. These are phasic motoneurons innervating fast muscle fibers. When a motoneuron is electrically stimulated, the normal order of recruitment is, therefore, inverted.

Also, it has been demonstrated that the concentration of collagen is higher for a slow than for a fast skeletal muscle. Collagen is the major connective tissue protein, having the main responsibility for mechanical strength of a muscle. More flexible connective tissue of a fast muscle allows faster movements, while slow muscle can store more elastic energy in its collagenous compartments.

When describing the state of skeletal muscle following spinal cord injury, it is important to distinguish between ***upper*** and ***lower motor neuron lesions***. The loss of force in electrically stimulated muscles paralyzed by an upper motor neuron lesion is due to disuse atrophy of the involved fibers. The state of the muscle following upper motor neuron lesion is, therefore, to some extent similar to the state occurring after immobilization. Immobilization of healthy human skeletal muscle seems to involve preferentially type I fibers. It causes a remarkable reduction in oxidative enzyme activity, whereas the glycolytic activity seems to be increased. Both immobilization and lesion of the spinal cord result in a decrease in metabolic demand. It was shown through animal experiments that all limb muscles are slow at birth, and that differentiation into the fast and slow types occurs during the first few weeks after birth. The differentiation of fast muscles is virtually unaffected by spinal cord transection, while the differentiation of slow muscles is greatly depressed. A few weeks after spinal cord lesion, predominantly slow muscles become nearly as fast as normal fast muscles.

The candidates for FES-aided locomotion activities are subjects with upper motor neuron lesions. Histochemical examination of spastic quadriceps muscle demonstrated type I fiber hypertrophy and type II fiber atrophy (decrease in single fiber area). On the contrary, a marked predominance of type II fibers was observed in subjects suffering from spinal cord lesion. Blood circulation also plays an important role when describing muscle properties. It has a direct influence on muscle fatigue. The nutritional blood flow in the paralyzed tibial muscle of paraplegic subjects was found to be significantly lower than in normal biceps muscle of the same subject (Kralj and Bajd, 1989).

1.5 ELECTRICAL STIMULATION METHODOLOGY

1.5.1 Stimulation parameters

Electrical stimulation is usually applied as a series of rectangular monophasic or biphasic (symmetrical or asymmetrical) electric pulses described by the following parameters:

- amplitude or intensity of pulses,
- frequency or pulse repetition rate,
- duration of single pulse,
- duration of a pulse train, and
- duty cycle.

The shape of the pulse train is an important parameter too, but it can be described using the basic parameters listed above (see Figure 1.5.1). In most cases of surface electrical stimulation applications, periodic monophasic or unidirectional pulses are used. Biphasic or bidirectional pulses prevent a slow deterioration of the electrodes, while the chemical conditions on the skin and in the neuromuscular tissue remain unchanged.

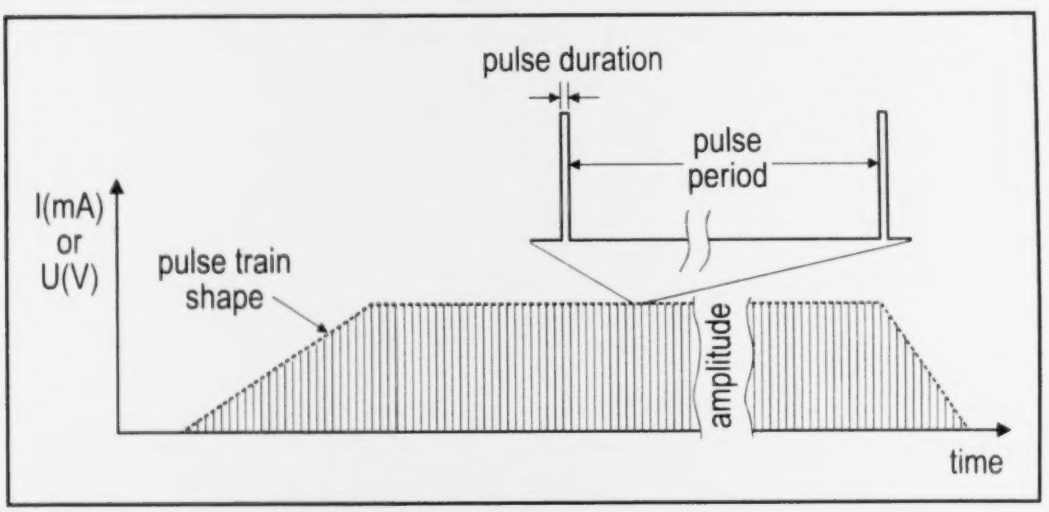

Figure 1.5.1. Stimulation parameters for unidirectional current or voltage stimulation.

Stimulating pulses are produced in the stimulator's output circuit, which can be either a constant-voltage or a constant-current generator. Constant-voltage stimulators provide a desired voltage, irrespective of resistance changes in the stimulated tissue. Constant-current stimulators generate pulses of constant current, also irrespective of resistance changes. An important difference between the two types of stimulator becomes evident in the case of an improper contact between the electrode and the skin. In the case of a constant-current stimulator, if the electrode-skin

contact changes, resulting in reduction of the effective electrode surface, the current density will increase, which may cause skin burns. In the same situation, using a constant-voltage stimulator, after the resistance increases, due to an insufficient contact, the current will decrease and there will not be any skin damage. Of course, the muscle response will decrease as well, which may be dangerous in unbalanced applications such as walking.

The joint torque is not linearly dependent on the stimulation intensity (Kralj and Bajd, 1989). There occurs at least two nonlinearities, i.e., threshold and saturation. The increase in joint torque due to an increasing amplitude of electrical stimulation occurs as a result of activating new fibers in a nerve bundle laying in an electric field between the electrodes. The main reason why all nerve fibers do not react to the same stimulation amplitude is found in the differences in the stimulation threshold and various distances from the stimulation electrodes. First, the fibers closest to the electrodes are stimulated. In addition, the fibers with a greater diameter respond earlier. From a certain stimulation intensity onward, the contraction force no longer increases. At such a stimulation amplitude, all alpha motor nerve fibers are excited, and a further increasing of the stimulus does not increase contraction. In surface stimulation of knee extensors , the values of the stimulation threshold range between 20 and 60 V , while the saturation value is between 100 and 150 V.

A single stimulation pulse provokes merely a short-lived muscle *twitch* of no more than 0.2 s of duration. If electrical stimuli are repeated every second, a twitch occurs every second, and, the muscle relaxes in the meantime. If the frequency of stimulation pulses increases up to 10 pulses per second (10 Hz), between two twitches there is no time left for muscle relaxation. Variation in force is considerably reduced at stimulation frequencies between 15 and 20 Hz. At higher frequencies , the response is already smooth; this is known as *tetanic contraction*. The frequency at which tetanic contraction occurs is called *the fusion frequency*. This frequency is not the same for all muscles and depends on properties of the muscle fibers.

Changes in *stimulation frequency* also affect the intensity of the response. The faster the stimulation frequency, the higher the joint torque is for the same stimulation amplitude. This is particularly expressed at lower frequencies, while changing the frequency between 40 and 100 Hz causes small differences in the isometric torque measured at the joint. A low stimulation frequency results in a less pronounced fatigue of the neuromuscular system. An electrically stimulated muscle fatigues more quickly than in the case of voluntary contraction. By electrical stimulation, the same nerve fibers are stimulated all the time, whereas with a healthy muscle the work is divided among different motor units of the same muscle. Due to a high stimulation frequency, the transmitter in the neuromuscular junctions is being exhausted, so the stimulated muscle soon shows signs of fatigue.

In a similar way to the amplitude, *pulse duration* also has a direct effect on the intensity of contraction. Here again, the response shows threshold and saturation features. The stimulation pulses longer than 0.7 ms not only do not produce a larger response, but, when applied through surface electrodes they produce an unpleasant sensation or even skin damage in subjects with incomplete spinal cord injury. Longer pulses also produce faster muscle fatigue. Therefore, short duration of stimulation pulses is preferred, while the force of a paralyzed muscle can be controlled by increasing pulse amplitude.

A functional movement of a paralyzed limb can not be obtained by a single electric pulse, but rather, by a series of stimuli of a certain duration, following one another at an appropriate frequency. Such a series of stimuli is called a *stimulation pulse train*. The relation of a pulse train duration and a pause between two pulse trains is often called the *duty cycle*, which, as another parameter of the stimulation, also affects the stimulated muscle fatigue. As expected, muscle fatigue is faster with larger duty cycles, particularly for higher stimulation frequency and shorter pulse trains (pauses).

1.5.2 Electrodes

The electrical stimulation can be applied through three different types of electrodes:

Surface electrodes are the most often used electrodes, particularly while evaluating patients for FES and the exercise phase of their rehabilitation. They are installed on the skin above the target nerve trunks or motor points of target muscle(s) and are directly wired to the output stage of the stimulator. Their advantages are as follows: installation is non-invasive and they are easy to install and remove, even by the subject, unless she or he has severely impaired hand dexterity; electrodes usually do not cause any skin problems; some types can stay attached to the skin for longer periods of time (weeks); and they are inexpensive in short-term use. Their disadvantages are: stimulation of cutaneous receptors causes triggering of unwanted reflexes; they have relatively low selectivity and low positioning reproducibility; skin irritation is not unusual; and their cosmesis may be inappropriate for some types of clothes.

Percutaneous electrodes are more and more in use as a transition from surface to fully implanted stimulation. Depending on their physical characteristics and the target site of stimulation, they come in four basic forms: uninsulated wire-endings, nerve cuff-electrodes, epimysial and epineural electrodes, and silicone-based electrode arrays. Independent of the form used, their active endings are implanted as close as possible to the target nerves or motor point(s) of the target muscle. The wires leading from the stimulating sites are usually collected in one or more convenient places under the skin, where they are taken out of the body. There, the wires are attached to a connector which enables the user to connect or disconnect the stimulator when desirable. Advantages of this type of electrodes are very high selectivity and the fact that electrodes can stay implanted for unlimited time. Disadvantages are: invasive technique of implantation; wire leads tend to break after some time, especially if they are led across the joints; they are difficult to service or to remove; the place where they emerge through the skin requires special maintenance and is a potential place of infection.

Implanted electrodes are most probably the future of long-term FES applications. They come in the same forms as the percutaneous electrodes plus another form of electrodes built into stimulating devices known as 'microstimulators' (Loeb et al., 1991). Similarly to the percutaneous electrodes, they are implanted as close as possible to the target nerves or motor point(s) of the target muscle. Electrodes have to be wired to a receiver (transceiver) implanted under the skin, which obtains commands and power from the control transmitter (transceiver) located outside the body. The control transmitter and the receiver communicate via coupled antennas. Microstimulators are different in a sense that they communicate directly with the control transmitter, since they contain all electronics required to receive and decode control commands and to perform the stimulation. Furthermore, they are relatively simple to install using a hypodermic needle. Advantages of the implanted electrodes are the same as those listed for percutaneous electrodes. However, there are fewer disadvantages, because no wires come through the skin, which reduces the risk of infection.

Microstimulators resolve most of the problems existing with percutaneous electrodes, but they introduce one new problem - the devices have to be enclosed in an electromagnetic field, otherwise the communication and power supply link breaks. This can be resolved by wearing a suit which has an antenna built in those parts which come above implanted microstimulators. The weight and size of such an antenna may still be a limiting factors for use of such devices in the lower extremities. However, applications in upper extremities are already achievable.

1.6 APPLICATION AND CONTROL OF FES IN LOCOMOTION

The problem of designing a motor neural prostheses could be compared to repairing a biological system which does not have accompanying 'technical documentation', as technical systems do. It is complicated by our lack of detailed understanding of the original 'equipment', and by our inability to use the same technologies. In most respects, the communication, stimulation, sensor

signal processing, and controller hardware available to contemporary engineers is quite primitive when compared to biological components. In addition, details of the methods of biological control are only partially understood. The biomechanical subsystems to be controlled are inherently nonlinear and possess time-varying input-output properties. For this reason, the use of nonlinear control methods and adaptive control techniques is of interest (Chizeck H.J., 1992).

After a series of electrical stimulators proposed and patented by different authors in the 1950s (Reswick, 1973), the first effort to apply electrical stimulation as an aid to recover function in a disabled person were made by Keegan Jr. (Reswick, 1973) and Liberson et al. (1961). They used electrical stimulation to help stroke subjects having a 'foot-drop' problem. Independently each from other, they developed a simple controller which detected lift of the heel of a disabled leg and delivered stimulation pulses to activate ankle flexion muscles producing ankle dorsiflexion during the swing phase of walking.

Alyeyev L. and Bounimovich S.G. were the first to attempt to use multichannel stimulation in achieving various functional movements in paralyzed people. The work, which is more interesting for its original idea than for the results achieved, is described by Reswick (1973) in his review article[1]. To date no controlled study has been reported to confirm their assertions. Their work was also the first attempt to try to record control signals from a 'normal' and apply them to a disabled person. No wonder that there was no more recent control study to confirm these results, bearing in mind all the morphological and structural changes of neuromuscular system that follow the

---

[1] "It was also at about the same time in 1965 that L. Alyeyev and S.G. Bounimovich reported some interesting developments in electrical stimulation from the Institute of Cybernetics in Kiev. Alyeyev had developed a multichannel stimulating machine which he had called Miotone. He claimed it operated on the following principle: Myoelectric signals of muscles used in various functions previously recorded on magnetic tape obtained from 'donors' were used to modulate up to six channels of an electrical stimulating machine to produce stimulation signals which were controlled by the original EMG patterns. He claimed to be able to produce functional movements in various types of paralysis in subjects. These functional movements produced by electricity not only caused muscle hypertrophy and improved metabolic function but also assisted in the reorganization of the central nervous system so that stroke subjects were able to regain control of various paralyzed functions."

spinal cord or brain injury resulting in paralysis. Even if the changes in the neuromuscular system were not that significant, the complexity of human gait makes the analysis of the gait also very complex, as it is reported by Braune and Fisher (1987)[2].

Development of FES applications usually results in electronic devices which can be generalized in the form of a so called ***NEUROPROSTHETIC DEVICE*** *(ND)*. The general structure and the role of the ND is illustrated in Figure 1.6.1. The ND has a modular structure built around its major module, the ***STIMULATOR***, which generates stimulating pulses and delivers them through the ***ELECTRODES*** to the nerves innervating muscles that have to be stimulated. The ***USER - ND INTERFACE*** module is used to communicate both the user's control commands to the ND and the feedback information from the ND to the user. This module enables the user (subject or therapist) to switch the ND ON or OFF, to set or modify parameters of the ND, and to override the decisions made by ND if required. It also informs the user about the status of the ND and about the events that are going to take place in the near future. ***COMMANDS*** provided by the subject allow for selection of desired motor tasks and scaling of effort (e.g., walking speed, height of step, grasping force). These commands are converted into electrical stimulation (Kralj and Bajd, 1989), or signals to active brace components as in Tomovic et al. (1972), and Durfee and Hausdorff (1990). They usually require voluntary action (e.g., finger-mounted switches and joysticks (Marsolais et al., 1985), or a shoulder-motion activated switch (Peckham and Keith, 1992)).

---

[2] "To complete our investigations we undertook to ascertain, as accurately as possible, the movements involved in walking using photography and all other means at our disposal.

Only after many tedious and sometimes disappointing preliminary experiments, in which Prof. Braune was tireless as always when a new method of research had to be elaborated, did we believe we had found a pattern in the experiments from which we could draw sufficiently accurate results. The experiments themselves were very time-consuming and fatiguing. From ten to twelve hours of uninterrupted activity were often necessary since preparing the experimental subject required the utmost care, as did the accurate arranging and insulating the Geissler tubes. Decisive experiments had to be carried out at night because there was no means of darkening the room in which we performed the studies.

The data resulting from the experiments permitted the transposition of the process of movement into a system of tridimensional rectangular co-ordinates. The calculations involved were voluminous and required continuous work for several months."

The interface modules and protocols should be designed to require as little conscious effort as possible, and to permit rapid and reliable command transmission. Feedback information could be presented to the subject in many forms. The simplest is to use audio/visual methods and present the information on a display together with the auditory information. Information could be either coded or explicit. More sophisticated and closer to natural sensory feedback is the use of substitutional sensory cues which can provide the user of the neuroprosthetic system with the information of the control state (e.g., ON/OFF, lock grasp type) or of the input or output state. Van Doren and Riso (1991) recently provided a review of the state of sensory feedback. Among more successful sensory substitutions is the implanted system used at Case Western Reserve University and the VA Medical Center in Cleveland (Peckham and Keith, 1992). Their system provides both state information and simple proportional information for grasp control by stimulating skin afferents without exciting underlying muscles through an implanted epimysial electrode on the shoulder or chest wall.

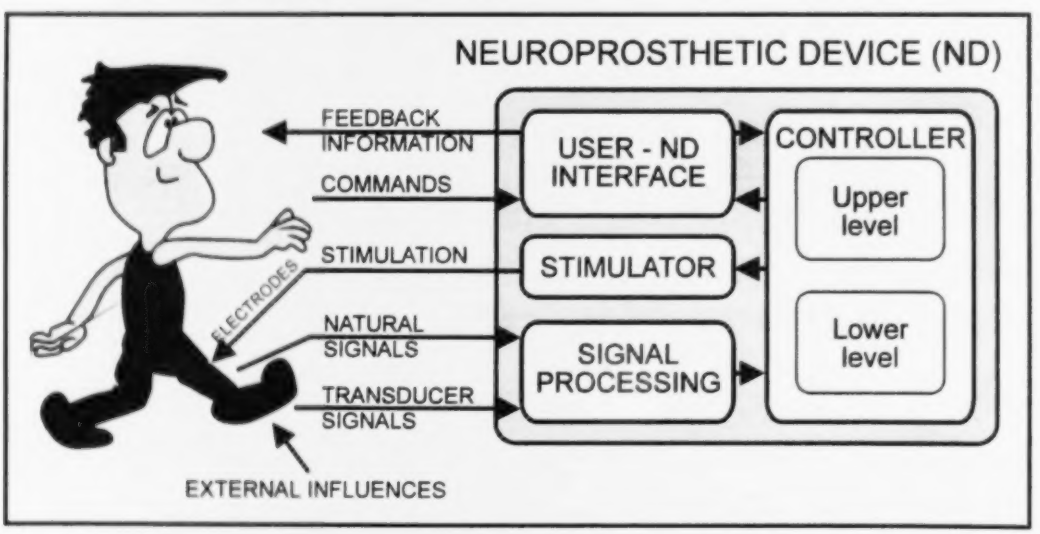

Figure 1.6.1. Modular structure of the Neuroprosthetic Device (ND) and its interface with the user.

An important tool for the improvement of the performance of functional electrical stimulation technology is the use of sensor-derived signals to modulate the electrical stimulation of muscles. Various recordings of either natural signals or biomechanical measurements are used as inputs to the ND. The purpose of recording signals from the biomechanical system involved in walking could be either to monitor the performance of the subject while walking or to use them as sensory feedback information in the control system for walking. Whatever the source of the signals, if these signals are intended to interfere with the control decision making process, the ND must have a ***SIGNAL PROCESSING*** module, which processes the signals, extracts the useful information from them and presents it to the decision making module. Depending on the type of signal, this module may be used for amplification, filtering, arithmetic modifications or delaying the signal. The module might be realized in hardware only, which provides the fastest processing and is now commercially available as digital signal processing (DSP) integrated electronic components, or in hardware and software, which, with recent extremely fast developments in high-speed, low-power microprocessor technology, became very competitive to the hardware-only solution.

Ideally, signals originating in natural sensors *(**NATURAL SIGNALS**)* might be used by the ND, just as they are used by the biological control mechanisms of intact muscles. Important steps in the development of the technology to bring signals recorded from natural sources (sensory nerves) closer to the reality are presented in Hoffer and Haugland (1992), Haugland and Hoffer (1994a, 1994b), Haugland et al. (1994), Popovic and Raspopovic (1992). The evaluation and implementation of neural signals for control purposes is reported by Sinkjær et al. (1992) and Popovic et al. (1993). Another source of natural signals are muscles. The use of EMG signals from above-lesion muscles for control of FES-assisted walking in paraplegics was demonstrated by Graupe (1975, 1983). Advanced technique which uses both, above-lesion EMG for timing control of FES, and below-lesion response-EMG for FES level adjustment was reported by Graupe (1988, 1989). At about the same time the potential of the myoelectric signal of electrically

stimulated muscle during recruitment in a closed-loop scheme was reported by Solomonow et al (1988).

To date, in practical applications only traditional sensors are used as a sensory feedback signal source *(**TRANDSUCER SIGNALS**)* for control purposes. Sensors that are most frequently used with ND include transducers of contact forces, slippage, joint angles, acceleration, and proximity. These sensors represent the reduced set of sensors and measurement equipment for kinematic and kinetic measurements presented in the section 1.7. A review of traditional sensors required for feedback control of neural prostheses appears in Crago et al. (1986), Payne (1989) and Webster (1992).

One of the most important parts of the ND is the ***CONTROLLER***. It is the 'brain' of the system, and it may be realized either in hardware or in a combination of computer hardware and software. It integrates and coordinates the functions of the other components of the man-machine (subject-ND) system. It takes inputs from the modules *USER - ND INTERFACE* and *SIGNAL PROCESSING*, applies its algorithm(s), and produces control outputs which drive the *STIMULATOR*. It also informs the user about its status of operation and about future actions through the feedback information channel of the *USER - ND INTERFACE*.

The control algorithm(s) can be either fixed or adaptive. The development of improved FES assistive devices requires designs that are either robust to modeling and parameter errors, or that adapt to them. The use of feedback control provides enhanced repeatability and predictability of muscle responses, and endows a neural prosthesis with the ability to adjust to both external disturbances (such as changing loads on limbs) and internal time variations (such as fatigue). The nonlinear and time-varying nature of electrically stimulated muscle (and the associated biomechanical system), as well as the discrete-event nature of many lower extremity functional tasks, provide significant challenges to controller design. A particular complicating factor is that

accurate predictive mathematical models of stimulated muscle and biomechanical interactions are unavailable for each individual impairment.

Basic research results in neurophysiology suggest a hierarchical structure of natural motor control in vertebrates (Prochazka 1993). This scheme can be crudely approximated to the control of FES-assisted movements. The artificial control of FES should consist of at least two major parts: the upper level controller (***coordination level***), which should make decisions on which movements to perform to fulfill a certain task, and the lower level controller (***actuator level***), which should initiate the required actions to perform a particular movement. The controller module of the ND shown in Figure 1.6.1 follows this basic structure. Of course, there is always a third level in movement control hierarchy - a voluntary control. It is not shown explicitly because of the obvious task to design a machine which will be supervised by the user and will serve its user. There are multiple bidirectional interactions between the user and the machine. In the presented subject-ND system, the subject's control of the unimpaired body parts via intact neural connections is assumed to be functional. The ***EXTERNAL INFLUENCES*** present unpredictable inputs to the system and both the user and the machine should be equipped to react safely.

To improve the effects of FES systems for locomotion, the combined use of a mechanical brace (orthosis) and FES for gait restoration was suggested (Tomovic at al., 1972; Andrews and Bajd, 1984; Schwirtlich and Popovic, 1984; Andrews and Baxendale, 1986; McClelland et al., 1987; Popovic et al., 1989; Solomonow et al., 1992). These systems are known as hybrid assistive systems (HAS) and may be used to provide better functionality of the ND or to add new features to it. There is a synergy between electrical stimulation, orthoses, and feedback control. One can think of both orthoses and feedback control as reducing the amount of electrical stimulation that is required to perform certain tasks. Feedback modulation of stimulation can accomplish this reduction by using less stimulation when there is less need (i.e., when the system outputs are close to target values). Insensitivity to external influences can be attained through adjustment of stimulation levels, rather than by excessive preset stimulation. Orthoses can reduce the

stimulation required by physically constraining certain motions, in order to provide biomechanical stability and support (Andrews et al., 1988; Popovic et al., 1989). Alternately, electrical stimulation can be thought of as providing motors for hybrid orthoses, with feedback-controlled stimulation yielding more precise torque generation. In addition, bracing can provide stable mounting sites for various sensors, allowing for improved measurements and thus enhanced feedback control effectiveness.

1.6.1 Control methods for a neural prosthesis

Control methods for a neural prosthesis are reviewed in Hollerbach and Bennett (1992) and Tashman and Zajac (1992). The configuration of the musculoskeletal system ***(plant)*** to control at any instant of time comprises the plant ***states*** (e.g., body segment positions and velocities). The devices that power the system are called ***actuators*** (e.g., muscles). The signals driving the actuators are the ***controls*** (e.g., neural pulse trains in intact humans and electrical pulses in FES systems). ***Controller*** is the processor which generates control signals (e.g., the CNS in the intact human, and the stimulus pulse train controller in the FES system). The time histories of the plant states in response to the control signals is referred to as the system ***trajectory*** (e.g., the joint-angle time histories). Terms defined above refer to the types of control systems presented in Figure 1.6.2.

Controllers can be designed to function without feedback sensory information and therefore without knowledge of the actual plant trajectory (see Figure 1.6.2(A)). This type of control is usually called ***open-loop control***. *Reference-based* open-loop controllers precompute and store (prior to task execution) muscle stimulation patterns that will, hopefully, execute the desired motor task. Muscle stimulation patterns are developed by combining clinical experience with trial-and-error methods, and then stored in the controller. Finding muscle stimulation patterns by trial-and-error which generate a smooth, energy efficient gait is difficult because of highly individual characteristics of the dynamic interactions among the body segments.

Muscle stimulation patterns can also be found by mathematically modeling the plant (i.e., musculoskeletal system) and the tasks to be executed. After the model is designed, computer algorithms are used to find stimulation patterns which will generate trajectories believed to fulfill the motor task requirements. One such algorithm uses ***inverse dynamics*** to calculate muscle stimulation patterns from an inverse of the musculoskeletal system model that will produce the torques needed for a set of trajectories to accomplish the desired motor task. Muscle stimulation patterns may be determined by inverse dynamics prior to task execution and stored in a reference-based open-loop controller.

Regardless of the design method and implementation, the performance of any open-loop control system will probably be inadequate, since *external influences* will cause performance to deviate significantly from that required for proper task execution. An external influence is any unexpected condition or event encountered by the plant. Functional neuromuscular stimulation systems are likely to encounter such influences; for example, from walking on uneven, inclined, or rough surfaces; from forces due to voluntary arm movements or wind; and from unexpected actuator performance, such as may arise from muscle fatigue. Even in the absence of these influences, open-loop control will probably be inadequate, since musculoskeletal properties will never be fully understood or perfectly modeled.

To correct for external influences and musculoskeletal modeling errors, a ***feedback controller*** with ongoing knowledge of the effects of the influence must be designed. ***Sensors*** (such as, joint-angle or ground reaction force sensors) provide measurements from which the current state of the system can be estimated. These estimates are fed back to the controller, resulting in a feedback, or ***closed-loop control*** system (Figure 1.6.2(B)). Closed-loop controllers are sometimes referred to as error-driven, since they respond to the trajectory error, determined by subtracting the measured trajectory from the desired trajectory. Error-driven systems are particularly well suited to tasks where the desired trajectories are constant or slowly changing, such as the maintenance

of vertical posture in standing (Stanic and Trnkoczy, 1974; Petrofsky et al., 1984; Wilhere et al., 1985; Jaeger, 1986; Mulder et al., 1987).

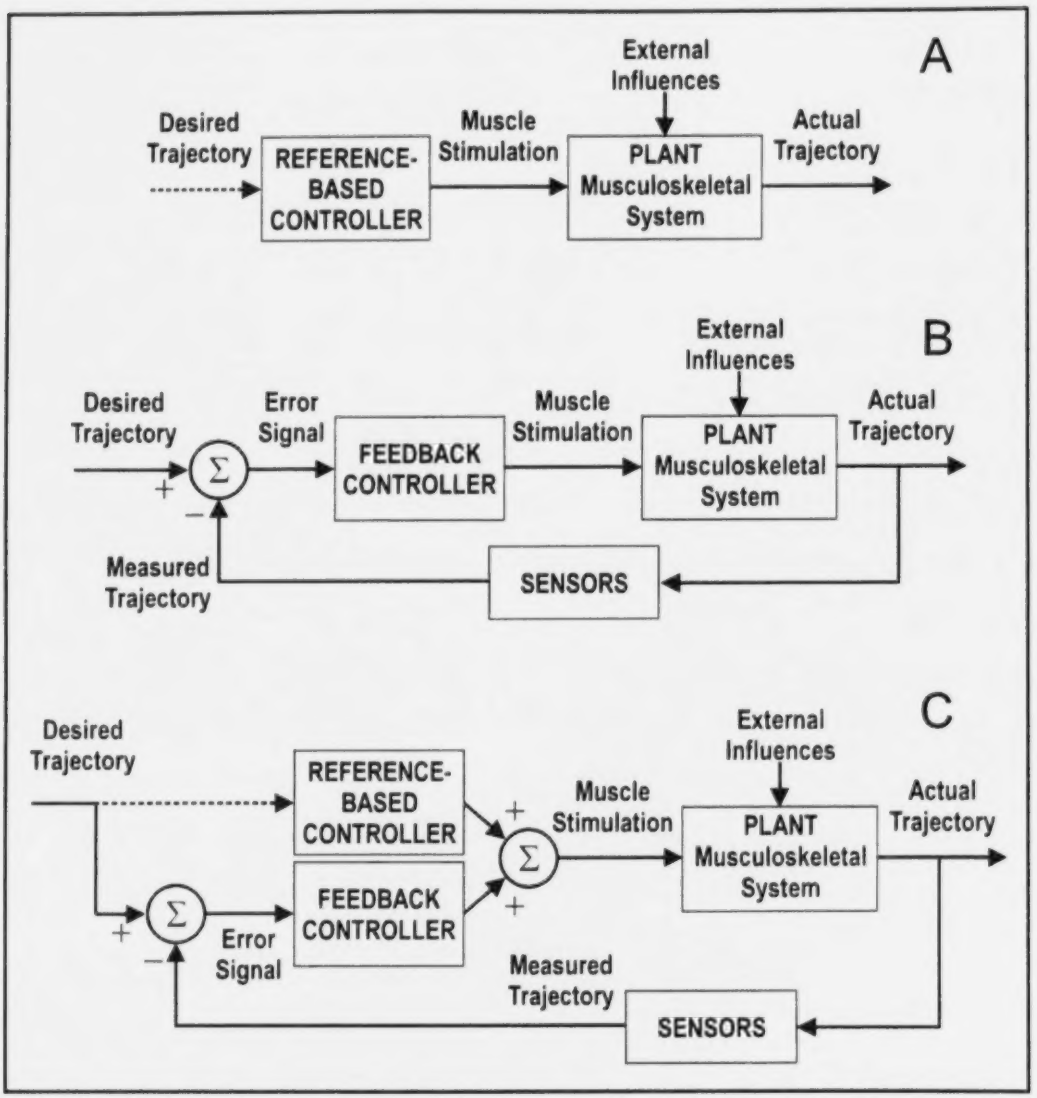

Figure 1.6.2. Various types of control systems: (A) Open-Loop Control (Reference-based); (B) Closed-Loop Control (Error-Driven); (C) Combined Reference-Based Closed-Loop Control (Error-Driven). The desired trajectory may be required for the design of the reference-based controller, but does not act as an input during controller operation; this relationship is indicated by the dashed lines in (A) and (C) (modified from Tashman and Zajac, 1992).

To control tasks where trajectories must change rapidly but predictably (such as walking), reference control is often employed in addition to error-driven feedback (Figure 1.6.2(C)). These controllers utilize open-loop control to generate an approximate trajectory (Figure 1.6.2(A)). A feedback controller similar to that shown in Figure 1.6.2.(B) is employed in-parallel to correct for trajectory errors resulting from external influences and modeling errors. If the body deviates too far from the desired trajectory, however, the stored controls may actually interfere with the recovery being provided by the feedback controller. This situation can arise because the reference-based controller functions independently from the feedback controller and cannot adjust its stored stimulation patterns during the task. Recent findings in neurophysiology suggest that animal motor systems may be organized in a very similar way, involving a preset motor program (Prochazka, 1993). Although it may look too simple to expect the natural control systems to operate without feedback, the equivalent of a motor control program is found to reside in the so-called 'central pattern generator' in the spinal cord of many vertebrates. Coordinated burst of activity may be recorded in muscle nerves even after the spinal cord has been isolated from descending input from supraspinal areas and from all modulated sensory input (Grillner and Zangger, 1975).

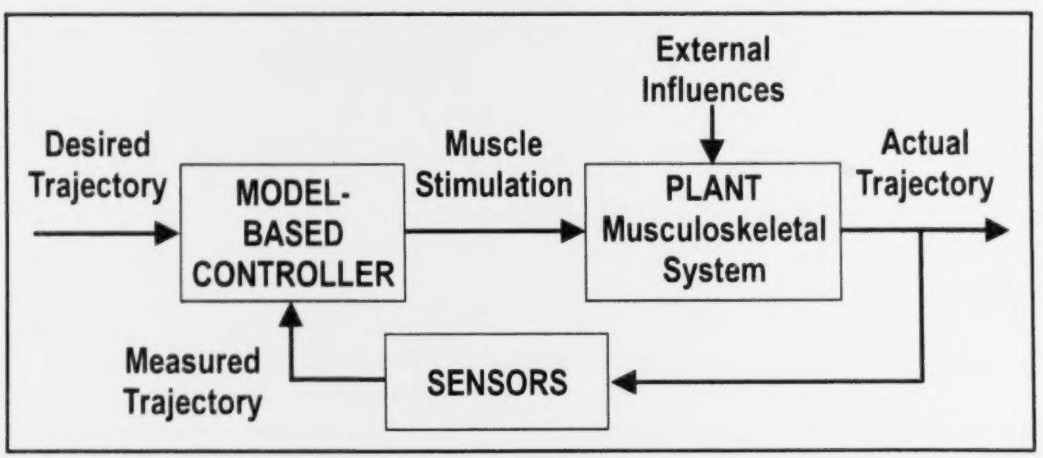

Figure 1.6.3. The model-based control system (modified from Tashman and Zajac, 1992).

A more robust system for walking could be developed with ***model-based control*** (see Figure 1.6.3). Model-based controllers utilize sensors and a dynamic model of the system to continuously recalculate the best trajectory to complete the desired task. During walking, if the body is following the desired path, the model-based controller generates the same muscle stimulation pattern as a reference-based controller designed with inverse dynamics. If the body deviates from the desired path, the model-based controller generates a new muscle stimulation pattern which, according to the dynamic model, should maintain or restore stable walking. Though this sounds attractive, the complexity and computational requirements of model-based control continue to limit its application to the laboratory environment or simple control systems.

Closed-loop control has many advantages for FES systems. A closed-loop control system can automatically compensate for small disturbances and consequent trajectory errors. Closed-loop control can also regulate system behavior more precisely than open-loop control, for example, by smoothing the motion during walking and reducing the amount of stimulation (Marsolais and Edwards, 1988). Feedback can also be used to detect potential falls, and either act to compensate and correct the condition, or shift to a 'safety' mode to reduce the chance of injury. In summary, closed-loop control lets the designer develop systems with a level of performance and flexibility well beyond that possible with open-loop control.

However, closed-loop FES control systems require sensors and are more complex to design and implement. Improper design can lead to instability, resulting in excessive stimulation and unpredictable (and potentially dangerous) performance.

Complex systems (such as the human musculoskeletal system) generally require ***dynamic models*** for control system design. A dynamic model is a set of equations that emulates the behavior of the physical system. Unlike static models, a dynamic model specifies how the current states of the system (e.g., body trajectories) depend on the past (e.g., the prior body trajectories and muscle forces). If the inputs to a dynamic model of a human motor task are the muscle

stimulation patterns, then the outputs could be the computed body trajectories. In control systems, dynamic model is used inverted ***(inverse dynamics)***, so that it determines the muscle stimulation pattern that will produce a desired trajectory. Although the resulting stimulation pattern could theoretically drive the system along the desired trajectory in the absence of external influences, some degree of modeling error is unavoidable, since dynamic models can never perfectly represent physical system. This error can be overcome by using error-driven feedback controllers in combination with reference-based or feed-forward controllers, such as one illustrated in Figure 1.6.2(C).

1.6.2 The importance of system modeling

In designing a neural prosthesis, the engineer attempts to develop a mathematical description of the system to predict its mechanical behavior under a variety of specified inputs and external loading. This is an analysis problem whose success at prediction depends on the accuracy of the mathematical model. The engineer will also use models during the process of control systems design and testing through simulation. A definition of what a model is and the purpose of modeling is explained in an overview of muscle modeling by Zahalak (1992). A model is a reduced or truncated representation of reality, and the two most important purposes of modeling are that a model can promote an understanding of its object and, further, it can predict the behavior of that object. A good model is one which is as simple as possible while simultaneously retaining the essential characteristics of the thing it represents. In his attempt to bring more order to the classification of muscle models, Zahalak distinguished between: *whole-muscle*, or *macroscopic Hill-type* models, that are based on the original work of A.V. Hill; *microscopic*, or *cross-bridge* models, following from the more detailed knowledge in structure from A.F. Huxley; fiber models, recognizing that series sarcomeres in a chain may differ in their properties; and the *distribution-moment*, or *intermediate* models that he himself worked on.

The importance of modeling in neural prostheses and methods to identify the model in neural prosthesis systems are reviewed by Durfee (1992). There are two types of models for predicting the behavior of a given physical system:

- ***parametric*** - whose structure is fixed, while the parameters of the equations are free to be adjusted to fit the particular system of interest; and
- ***non-parametric*** - where no assumptions are made about the structure of the system. Instead, the system is viewed as a 'black box' described by one or more functions which completely characterize the input-output behavior. Non-parametric models are also known as 'black-box' models or 'functional' models.

To design a controller for a motor neural prosthesis, one must develop an accurate model of the musculoskeletal system, the 'plant' to be controlled. The plant to control consists of the skeletal links, the joints, and the musculotendon actuators. A state of the art in musculoskeletal system modeling appears in Winters and Woo (1990) with a survey of the human musculotendon actuator parameters by Yamaguchi et al. (1990).

The control of walking is more difficult than the stabilization of a multisegment inverted pendulum. The gait cycle consists of distinct stance and swing phases, which are inextricably linked. Furthermore, the joints move through ranges of motion and positions that may prohibit linearization. A feasibility study done by Yamaguchi and Zajac (1990) demonstrated the challenge of modeling walking and designing controllers for FES-assisted walking. One of the results obtained was that some of the muscles (e.g., the plantarflexors) need to produce forces in excess of 50% of normal strength, which exceeds the current force generation capacity of FES-reconditioned paralyzed muscle (Kralj and Bajd, 1989).

1.6.3 Finite-state models and its derivatives

One way to design a ***non-parametric model*** of a controlled system is to analyze its inputs and outputs and to tailor the rules of its behavior. In a paper, that is considered a very important

milestone in the field of control, Tomovic and McGhee (1966) have introduced a ***finite-state approach*** to the synthesis of control systems. Since the proposed design in this thesis is based on the principles introduced in this article, that particular work will be presented here in more detail. Tomovic and McGhee demonstrated the use of a mathematical theory of ***finite-state automata*** in analysis and synthesis of simple control devices for bioengineering systems. In order to provide a connection between automata theory, which is basically concerned with discrete decisions, and control theory, which deals with the dynamics of continuous systems, the concept of a ***cybernetic actuator*** was introduced. It represents a mechanical analogy with the natural actuator in biological systems - muscle, and it provides any mechanical system, which attempts to duplicate the functioning of a limb, with four basic actions (states) observed in natural limbs in complex motions: rotation of the joint in either direction, locking the joint and unlocking the joint so that it can swing freely. Two binary inputs are enough to define four output states of the cybernetic actuator. When used as an element of a bioengineering system, it is to be expected that a cybernetic actuator would normally rotate the joint between two fixed limits. If z denotes the output of the actuator and z_a and z_b denote the two limiting values for z, then the reachable set of points S for the actuator is defined as a closed interval $S = [z_a, z_b]$. It is of the greatest importance to realize that while the input to a cybernetic actuator may assume only four distinct states, the output may be driven to any point in the continuum forming the reachable set and, if desired, locked at that point. This property of a cybernetic actuator permits the link of control theory and automata theory. In addition to this basic description of the cybernetic actuator, one more variable is potentially available - the amount of energy stored in the actuator.

In order to permit full use of the capabilities of cybernetic actuators in bioengineering control systems, it is important to recognize a possible requirement for feedback signals. In many applications, such feedback may be provided by the sensory capabilities of the natural biological elements of the system. For example, in the FES-assisted movement of the paralyzed leg, with no

afferent information coming from the leg at the subject's conscious level, visual feedback can be used to replace the proprioceptive and cutaneous feedback available in a normal leg. On the other hand, it is often desirable to attempt to provide explicit feedback to decision networks in order to grant a degree of autonomy to the behavior of certain parts of a control system.

Feedback requirements for a particular application could be characterized by associating a second set of points with each actuator. This is called the *decision set* T, consisting of all those points in S from which information is fed back to the controller responsible for the state of the actuators. In conventional continuous control, the set T is equivalent to the set S. For finite-state control, T must be a finite subset of S. The elements of T are called *decision points*. Each decision point should use a sensor as a source of information for its input. The simplest general type of sensor, which provides a binary output from a continuous input is the *threshold gate*, or *comparator*. In its two-input form, a threshold gate is a combinational circuit element with two continuous inputs and one binary output. One of the inputs is designated as a *reference input*, and the other as the *signal input*. The action of the threshold element is such that the output assumes the *one state* when the signal exceeds the reference; otherwise, it assumes the *zero state*.

The system which consists of cybernetic actuators as power elements and decision elements with threshold feedback and binary memory was named an *actuator network*.

The question of how to measure the performance of the actuator networks was difficult to answer. It is well known that the optimization of conventional control systems is generally based upon some integral or functional performance criterion. On the other hand, in automata theory, optimization usually refers to the minimization of the number of states in a sequential machine or minimization of the number of elements in a combinational network. Since actuator networks involve both continuous motion and binary decisions, none of these criteria can be applied

directly. In order to reach a satisfactory system design, some sort of ill-defined compromise must be made. Optimal actuator network still remains undefined.

When an actuator network is to be integrated into a bioengineering system involving a human being, yet another important consideration arises. While capacity of a human being to make complex decisions is unmatched, and is basically responsible for the versatility of human systems, the rate at which humans are able to make conscious decisions is very low. This fact motivates the design of systems possessing the properties of maximal autonomy or minimal exchange of information between the conscious level of human decision making and lower levels of the system. Of course, every application has its own autonomy threshold. For example, if a person with a complete lesion of the spinal cord, resulting in a paraplegia (paralysis of both legs), was provided with switches controlling eight channels of stimulation, it is doubtful that he would ever be able to manipulate the switches rapidly and accurately enough to achieve a satisfactory walk. This and similar problems are usually resolved by increasing the autonomy of the control system by using feedback sensory signals or by inserting or using existing time-dependence between control events (enabling a subject to control lists of events rather than particular events). The list of control events can be either artificial (based on biomechanical analysis of human locomotion), or natural, such as the sequence of movements resulting from activation of the flexor withdrawal reflex.

The theory presented above has been successfully applied to a problem of synthesis of a control system for an artificial leg. With two actuators at the knee and the ankle joints, eight threshold gates, processing feedback sensory signals from five sensors (heel contact, hip, knee, ankle and toe angles), were enough to define six different states. The design of this finite-state model was based on the comprehensive photographic studies of the natural leg behavior during steady walking.

Trying to resolve problems in designing finite-state control methods, Tomovic (1984) proposed a closed-loop, non-numerical, control method, called ***artificial reflex control***. Artificial reflex control refers to a skill-based expert system[3] using rules that have an *IF(...) THEN(...) ELSE(...)* structure (Andrews and Baxendale, 1986; Andrews et al., 1989; Popovic et al., 1991; Franken, 1994). This method of control belongs to a group of so-called ***rule-based control (RBC)*** methods, in which the cyclic locomotor activity is presented as a sequence of discrete events. A sensory pattern occurring during particular motor activity is recognized with the use of traditional and/or natural sensors. The specific discrete event is called the 'state of the system', by analogy to the state of a finite-state automata. A recognized sensory pattern during a specific state of the system initiates a corresponding functional movement. As a logical extension to this work, in 1987 ***adaptive reflex control*** was proposed (Tomovic, 1987).

Rule-based control with hand-crafted rules was applied in intrinsic FES systems and in hybrid assistive systems, comprising the FES and active or passive orthosis, but the problem of accurate and reliable control has not been solved yet.

1.6.4 The derivation of control rules for RBC

Control rules can be written by the researcher based on his or her previous experience. This method of rule definition is known as ***hand-crafting rules***. The quality of the resulting gait may then be assessed, adjustments performed and the process repeated in an iterative fashion until a satisfacory gait is achieved (Kobetic, 1994). This 'trial and error' method does not guarantee a formally optimal gait, and requires the subject's presence at every iterative cycle. Another disadvantage of this method is that the performance of the resulting control system depends on an expert's abilities to select and express his or her knowledge explicitly in rules.

---

[3]The term "skill" refers to sensory driven functional motions controlled at the execution level by reflexes and automatic actions acquired by learning and training. Control activities of this kind represent, thus, a form of knowledge which may be evoked without involvement of the voluntary level.

The transfer of human knowledge to the computer knowledge base relies basically on identification and representation of ***invariant features*** of functional motions. While the method works satisfactorily in modeling the gait of non-impaired persons, its application to handicapped persons is not as successful because each case of sensory-motor deficiency is in many ways specific. This may be the reason that, although, expert systems based on the transfer of human knowledge to the machine at the conscious level are in wide use today, machine control by skill-based AI (Artificial Intelligence) systems is still in the stage of early development. The most serious problem on the way to faster development of skill-based control system is the description of human skill activities in machine representable form. By definition, automatic features of motor skills are expressed as spatio-temporal events (Tomovic, 1989). Consequently, the capturing of such knowledge requires a new type of identification methods which rely to a large extent on external manifestations of neuromotor control mechanisms.

A complex movement can be improved with training, thus the acquisition of skills can be considered as an optimisation process taking place at a number of levels in the central nervous system. A skilled subject or therapist manually controlling FES-assisted locomotion can develop a set of motor control rules that are near optimal. If these rules can be copied and stored, they can be used to form a controller to reproduce the near optimal movements. This presents a basis for another approach to control rule definition: transfer of the knowledge, stored in form of a skill, from a skilled subject or therapist to a computer, using an automatic process that employs machine learning algorithms ***(automatic rule generation)***.

1.6.5 Automatic generation of control rules for RBC

In the case of very complex system (such as the walking of a subject with incomplete SCI) the feed-forward closed-loop control system with a machine-learned non-parametric model may be an alternative way to design a controller (see Figure 1.6.3). Instead of tailoring the rules for every subject, they may be generated much faster by machine learning programs (such as ***neural***

networks for supervised learning or an ***inductive learning*** algorithm) in the task of mapping the skilled functional performance of the subject or therapist. To be able to generate the rules or control decisions, the machine learning program has to be presented with input and output data samples representing the behavior of the system to be controlled. If the control task is to reproduce skilled switching of the FES during walking of a subject with incomplete SCI, the inputs to the machine learning program can be signals recorded from natural and/or traditional sensors, qualitative verbal information describing performed actions, or quantitative assessment of the same actions. The output can be single or multiple channel switching function of the FES required to perform these actions.

This was proposed for the upper level controller for FES-aided walking of subjects with incomplete spinal cord injury by Kirkwood and Andrews (1988, 1989) and Andrews et al. (1989). Being inspired by the "Boxes" algorithm implemented in the "pole balancing" paradigm by Michie and Chambers (1968), Kirkwood and Andrews evaluated the use of inductive learning technique and traditional transducers for automatic generation of control rules by transferring the skill of the subject with SCI in manually controlling a simple two-channel-per-leg FES-system for paraplegic walking, such as one described by Kralj and Bajd (1989). In a continuation of this work, Heller (1992) evaluated the use of inductive learning technique in controlling the swing-through walking of paraplegic subjects. Their results demonstrated possibility to induce rules that describe human movements by using inductive learning technique to find invariants in movement patterns. However, the question of a generalization from machine learning technique training to a real-time control application remained unanswered.

After preliminary results demonstrated possibility of using neural networks for designing control rules for FES-assisted walking of subjects with incomplete SCI (Stein et al., 1992; Kostov et al. 1992), a similar approach was adopted as the basis for this thesis project. In this approach to creating control rules for rule-based systems there are limitations which have to be considered,

the most important one being the limited capacity of machine learning technique for supervised learning to learn only those events which are presented to it during the training.

1.6.6 Related work

Related work is usually found in the field of robotics and automatic control. A method for the control of a class of nonlinear systems using feedback linearization, when the nonlinear models are not well known is described in Chizeck (1992). It evolved from the basic idea (Loparo and Teixeira, 1990), by which one neural network was used to learn the inverse dynamics of the nonlinear system to be controlled and the second one to learn the feedback control laws that make the system under control behave like a desired system. Veltink et al. (1990) have used a backpropagation multi-layer perceptron network for reconstructing muscle activation patterns in the walking cycle on the basis of signals recorded from external sensors (goniometers and foot-switches). Beckmann et al. (1992) have integrated a backpropagation neural network into a simulated control loop for a dynamic and nonlinear model of a paraplegic subject. Their computer simulations demonstrated that the proposed feedback-loop with an integrated neural network was capable of stable controlling the model system. Karsai (1991) reported on a series of experiments for controlling a nonlinear systems, such as industrial plants, using various neural networks in controller architectures. One of the proposals was to copy the function of an existing controller - technique similar to the one proposed in this thesis.

Kelly et al. (1990) have applied two neural networks implementations to myoelectric signal (MES) analysis tasks for development of more reliable methods of deriving control for multidegree of freedom arm prostheses. In the first implementation they have used a discrete Hopfield network to calculate the time series parameters for a moving average MES model. The second implementation involved using a two-layer perceptron for classifying a single site MES based on two features, specifically the first time series parameter, and the signal power. Using these features, the perceptron is trained to distinguish between four separate arm functions.

The preliminary results of an attempt to compare two approaches to the automatic generation of control rules (inductive learning and neural network approach) in the example of muscle activation modeling from kinematic variables measured during normal human walking are presented in Heller et al., (1993). The advantages of the rule-based inductive learning technique are: the rules are explicit, comprehensible, easily encoded into a knowledge base and may be executed quickly; and the algorithm is variable selective, i.e. it is possible to identify the most important attributes (derived from particular sensors) in any rules that are produced. An advantage of the neural network technique was that one network was able to model the output of more than one muscle. Main conclusion was that there is an overall advantage of these techniques over explicit inverse musculoskeletal models.

Considering flight control of an aircraft as one of the most challenging applications for neural networks, Schley et al. (1991) have integrated optimal rule-based control of nonlinear plants using backpropagation neural networks. They demonstrated an approach for a control task consisting of an aircraft flight path transition problem. Hruska et al. (1991) demonstrated the practicality of designing and training a neural network for a specific application with limited utilization of a human expert. They showed how learning algorithms developed for neural networks may be used to enhance the effectiveness of rule-based expert systems. The previous two reports also showed that there is no competitiveness between expert systems and neural networks, but by combining the two a superior performance compared to either approach alone will be obtained. The key to success is to use each system for what it does well; pattern recognition and rules generation by the neural network, and application of control rules by the expert system. Such a configuration of a control system is proposed in this thesis.

1.7 INITIAL THESIS HYPOTHESIS AND OBJECTIVES

1.7.1 The initial hypothesis tested in this thesis

Machine learning techniques can provide a non-parametric model for controlling FES systems for assisting walking in individuals with incomplete spinal cord injury, and when combined with hand-crafted restriction rules can achieve better performance than a hand-crafted rule-based system alone. In addition, machine learning techniques sucessfuly generalize the model derived from manually controlled to automatically controlled FES-assisted walking not only during the same walking session, but also to walking occurring several days after the training.

1.7.2 The proposed approach to modeling of the control system

In this thesis, the system to control is a walking subject with incomplete spinal cord injury. The voluntary control over the non-disabled functions is present and the goal is to design a better control system in the sense of higher accuracy and reliability. The integrated control system (ICS) for evaluation of walking of new subjects and design of control rules for FES-assisted walking is proposed. The ICS is based on machine learning techniques for supervised learning, such as Adaptive Logic Networks (ALN) (Armstrong et al., 1990, 1991), a type of artificial neural network. ALNs are implemented in form of computer program designed to learn how to approximate or map the functional relationship between its domain and codomain. If the set of natural feedback signals or signals from traditional sensors is representing a domain of a mapping and the set of possible output values that has to be mapped represents a codomain of the mapping, then the result of the ALN training is approximation of an input/output transfer function. This function is either a boolean expression that can be presented in the form of logic trees with boolean operators in its nodes or a decision tree. In feasibility testing, which is part of this thesis, different preprocessing methods for filtering information are evaluated and optimized for certain types of sensory and control

signals. Based on ALN learning and performance results, a source of the signals representing domain is selected for particular applications.

1.7.3 The specific reasearch questions

1. Which methods are available for control of FES for locomotion of subjects with SCI?

2. What are the physical requirements for different types of control?

3. Is manually controlled walking better or worse than a control system that automatically recognizes a subject's intentions?

4. Which measurements from the subject's body are available for recording?

5. How can these measurements be transformed into signals useful for description of the subject's state during walking?

6. Which machine learning methods are available, and which perform better in the task of mapping the transfer function?

7. How can machine learning be incorporated in rule-based control systems operating in real-time?

1.7.4 The major practical objectives

1. Identification and evaluation of locomotor problems experienced by subjects with incomplete spinal cord injury that will be targeted by this project.

2. Review of previous FES implementations in walking of people with spinal cord injury.

3. Investigate methods for the design of controllers that will allow a subject with incomplete SCI to walk using FES.

4. Experimental recording on subjects with incomplete SCI and experimental animals to acquire potential feedback signals.

5. Development of a hand-crafted rule-based system for control of the stimulation in both animal and human model.

6. Development of the graphics interface program for input and output signals to and from learning algorithms.

7. The evaluation of ALNs for control of FES based on the signals recorded either from natural or traditional sensors. ALNs will be tested as learning tools able to generalize from a training set of prerecorded data to test signals not used in training.

8. Comparison of the performance of ALNs and inductive learning techniques in transfer function mapping tasks.

9. Designing a safe FES control by optimization of ALN parameters and implementation of restriction rules for supervision of ALN control.

10. Development of software and hardware implementations of the ALN learning algorithm in rule-based systems for control of FES in walking.

11. Test experiments of the developed technique on subjects with incomplete SCI.

2. METHODS AND MATERIALS

2.1 SUBJECTS

2.1.1 Selection of candidates for FES-aided locomotion

In most of the clinical trials presented in this thesis we concentrated on selected individuals with incomplete spinal cord injury (SCI) affecting their ability to walk. The only exceptions are the results presented in the Section 3.4 which were obtained using signals recorded from a young subject (A.M.) with complete spinal cord injury. All were able to stand and walk with stimulation for a limited distance. In the feasibility and evaluation studies of different machine learning algorithms the data from seven SCI subjects were recorded during routine gait analysis, and stored for off-line processing. The information about their injuries, level of disability and current rehabilitation status is summarized in Table 2.1.1. All subjects signed the consent form approved by a local ethics committee.

Although the participation of all subjects deserves exactly the same appreciation, one subject was the most active participant in the whole study from the early stages of the manual control of FES for walking, up to the latest stage achieved in integrated control system development. Since most of the results presented in this thesis were obtained with that subject, she is presented here in more detail.

Table 2.1.1. Information about the subjects participating in the study

Subject	Age	Sex	Years post injury	Lesion level impairment	Major gait deficit	Orthosis	FES m. - muscle n. - nerve	FES control	Walking aids	Lesion cause
C.C.	27	M	8	C3/C4. Frankel D	right swing	right Ankle Foot Orthosis	right Quadriceps femoris m.	one manual switch	crutches	motor vehicle accident
N.D.	38	M	6	C7. Frankel C	right swing	left knee Orthosis	right Common Peroneal n.	one manual switch	crutches	fall
C.H.	37	M	6	C5/C6. Frankel D	left swing	none	left Common Peroneal n.	one heel switch	cane	diving accident
L.S.	57	F	5	T9. Frankel D	right swing	none	right Common Peroneal n.	one manual switch	four-point walker	transverse myelitis
M.W.	26	F	8	C5/C6. Frankel C	right stance and swing	none	right Common Peroneal n., right Quadriceps femoris m., right Gluteus maximus m., right Hamstrings m.	one manual switch	crutches	motor vehicle accident
L.W.	47	F	8	C2. Frankel C	left swing	none	left Common Peroneal n	one and two manual switches	four-point walker	arterio - venous malformation
A.M.	10	F	6	T2 complete paralysis	bilat stance and swing	bilateral Knee Ankle Foot Orthosis	bilateral Common Peroneal n	two hand switches	four-point walker	motor vehicle accident

2.1.2 Subject L.W.: Case History

At age of 39, in October 1986, the subject suddenly experienced spontaneous severe left temporal headache initially described as a pressure sensation accompanied by burning on the left side of her face and neck. This was the worst headache of her life so she came to the Emergency Department of the University Hospital. She had neck stiffness. Very soon she developed right arm numbness and weakness and this progressed to involve the left leg as well. Her previous medical history was unremarkable with the exception of a transient episode of numbness involving her right face and shoulder lasting one day. Neurological examination showed decreased sensation to pin prick more on the left than on the right. Position sense was normal on the right but impaired on the left foot. Power was normal on the right but less than anti-gravity in the left arm and leg. Tone was increased in both legs. Right arm tone was normal. The left arm was flaccid. There was sustained clonus bilaterally at the ankles.

After very extensive study which lasted a few months, she was diagnosed with hematoma within the cervical (C1/C2) segment of the spinal cord in association with a syrinx extending to the C5 level on one side. The Arnold-Chiari malformation was identified. Rehabilitation, which included very aggressive physiotherapy started in January 1987. She was discharged in May 1987 with the following diagnosis:

- cervical cord hematoma with syrinx leading to partial quadriparesis,
- Arnold-Chiari malformation,
- neurogenic bowel,
- neurogenic bladder.

She had been a very active person, working as a fitness instructor before the vascular incident. The disease has put her in a wheelchair, but not for long. With a functional disability described as follows she was in April 1989 included in the rehabilitation program at the Division of Neuroscience, University of Alberta:

- right hand - normal
- left hand - poor fine hand function but has grasp
- right leg - can swing through and can weight bear on it if it is braced at the knee
- left leg - can hold her weight if hyperextension is controlled with a Swedish knee cage but can not actively swing through to step
- standing - can attain and maintain on her own with four point wheeled walker or in parallel bars
- walking - since the subject has no voluntary hip and knee flexion in her left leg, walking without external stimulation of the left leg was not possible.
- She had a considerable degree of spasticity (grade 3 on the modified Ashworth scale) in both legs and occasional periods of clonus, particularly at her left ankle joint, despite taking baclofen (Lioresal, 80 mg/d) and cyproheptadine hydrochloride (Periactin, 16 mg/d).

During preliminary experiments with surface stimulation, it was found that the subject may benefit even from single-channel FES applied to the left common peroneal nerve, producing swing of her left leg. Walking was very slow (see Figure 3.1.1 A), but could be improved by adding extra channels of stimulation to the right quadriceps and the right gluteal muscles to stabilize the right knee and hip joints while the left leg went through the swing phase (see Figure 3.1.1 B). The extra channels of stimulation increased the speed of walking from 3.4 to 4.3 m/min., without changing the stride length. The very long stance phase was shortened from 5.7 to 3.9 s and the swing phase from 1.9 to 1.4 seconds (Stein et al. 1993). The steps also became more regular.

The subject's high motivation for walking and availability of appropriate technology resulted in surgical installation of single-channel fully-implanted stimulator on her left common peroneal nerve in popliteal fossa. The stimulator, a modified Mikrofes, designed and manufactured in Ljubljana, Slovenia (at that time republic of SFR Yugoslavia), provided external excitation to her left common peroneal nerve to elicit the flexor withdrawal reflex (Kandel et al., 1991) during swing phase of her left leg. The stimulator was implanted in May 1990 and very soon after the surgery, the subject

started using it for walking. At the same time, there was ongoing research to decide which type of FES control should be used with the stimulator to suit her walking needs the best. The stimulator was initially controlled manually by the researcher and later the subject has learned how to do it herself by pressing on the manual switch installed close to the right hand grip on the frame of her wheeled walker. Other stimulation sites were tested using surface electrodes with the goal to improve her posture and right knee extension. The switching control of multiple channel stimulation during walking was not an easy task. Thus, the practical system she continued to use in her daily routine walking remained the single-channel implant.

In early 1991 the control of her single-channel stimulation system was expanded by adding foot-switch control of stimulation timing, which automated the control by recognizing the weight bearing on the right, less disabled leg, and starting the stimulation after a certain delay. The new, automatic control not only resulted in improved range and faster walking, but it also provided her with protection against triggering of the stimulation by the clonus on both legs, which occurred regularly during almost every walking session. The operating principle of this controller is presented in Section 3.2.

In late 1993 the subject was implanted with percutaneous wires for multichannel FES distributed to stimulate the following sites:

- common peroneal nerve (induces ankle dorsiflexion and flexor withdrawal reflex),
- quadriceps muscle (knee extensor),
- psoas muscle (hip flexor),
- gluteus maximus muscle (hip extensor).

The goal was to install wire electrodes and prepare the stimulation sites for a fully implanted multichannel stimulator Pulsar (MiniMed, Sylmar, CA) which is still undergoing development for human use. The four channel stimulation preserved previously used functions of activating the flexor withdrawal reflex during the swing phase of the left leg, but also added control over left knee

extensor muscles during stance phase and control over left hip flexion/extension in appropriate phases of the gait cycle.

In parallel with participating in the development of better and more complete stimulation systems for walking, the subject participated in the research and development of a better and more reliable control system for FES-aided walking. She had gone through all phases of control system development, from a manual control system, to automatic switching control with discrete logic, a microprocessor-based finite-state control system, and eventually the integrated control system based on artificial neural networks.

2.2 EXPERIMENTAL ANIMAL

Some techniques were not ready for use on the human subjects and therefore we did a limited number of animal experiments which are presented in Sections 3.7.1 and 3.7.2. These experiments were performed on adult cats of either sex that were trained to walk on a powered treadmill in a range of walking speeds from 0.4 - 1.0 m/s. Triphasic cuff electrodes (Stein et al., 1977) were implanted around several of the following nerves: sciatic, superficial peroneal (SP), tibial (TI), common peroneal, and sural. Epimysial EMG electrodes were sewn to medial gastrocnemius (MG) and tibialis anterior (TA) muscles.

Surgical Procedure: Under fully sterile conditions, a gas-sterilized set of electrodes was implanted and a head and/or back connector was attached according to the following protocol approved by a local ethics committee. Prior to surgery the cat was injected intramuscularly with an antibiotic (Ayercillin) to minimize the risk of infection. It was then anaesthetized with 50 µg/kg of Somnitol given intraperitoneally. Regulation of anaesthesia depth was accomplished by regular monitoring of heart rate, respiration rate and the absence of reflexes to noxious stimuli (eye blink and limb withdrawal to paw pinch). Ketamine and atropine were injected intramuscularly and the cat was intubated to allow artificial ventilation, if necessary. During surgery blood pressure and

body temperature were monitored, and temperature was maintained constant with a heating pad. An intravenous catheter was inserted into the cephalic vein for infusing fluids when necessary. The cat was shaved and scrubbed with Betadine in the region of the incision before being draped with sterile sheets. These procedures were carried out by veterinary resource personnel.

After the surgery, the skin was closed with sutures and the tracheal tube and venous cannula were removed. When a back connector was used, the leads were brought out through the skin and attached to the connector after the surgery. After completing the procedures, the animals were transported to the intensive care unit of the animal care facility. Postsurgical care included one week of antibiotic therapy (Ayercillin), opoid-based analgesia (Buprenorphine) as required, and close supervision by investigators and support stuff in the animal care facilities of the University of Alberta.

Sensory signals were recorded from TI and SP nerves using cuff electrodes and EMG activity was recorded from MG and TA muscles using epimysial electrodes. These signals were used in evaluation of biological feedback signals for binary and analog control of the ankle flexor and extensor muscles.

2.3 SELECTION AND DESIGN OF SENSORS

The nonlinear response of the muscles to the electrical stimulation, muscle fatigue, pain, and spastic hyperreflexia are some of the problems that set the demand for feedback control (Lan et al. 1991, Chizeck, 1992). Feedback control is used in natural motor control systems, and in principle, could be used to resolve these problems, but it has many practical difficulties, mainly due to the inadequacy of available sensors (Prochazka, 1993). A review of sensors required for feedback control of neural prostheses appears in Crago et al. (1986), Payne (1989) and Webster (1992).

Selection of the appropriate sensors for FES control system design which includes feedback signals is one of the most difficult tasks in the beginning of a control system design. If the set of available sensors is limited only to those which can be used without significantly disturbing the subject's normal life, the number of useful variables that can be used as an information source for feedback control systems is drastically reduced. Position measurements, which are usually obtained by very complex video motion-analysis systems are obviously not suitable for any use outside a laboratory. Exclusion of such systems reduces the possible sources of feedback signals to those which are natural parts of the subject's body, such as nerves and muscles producing neural and myoelectric signals respectively, and to the traditional transducers that can be installed on the subject's body to measure body's trajectories or interaction forces between the subject's body and the environment.

In order to chose traditional sensors for control system design presented in this thesis, an extensive study was done on the topic and the sensors were selected based on the following criteria: usability, availability, cost, durability, signal reproducibility.

2.3.1 Studying walking

The pattern of walking of each person in as unique to that person as his or her personality. While each person is different, there are certain attributes of gait in healthy subjects that are quite consistent or consistent within a 'normal' range. The same attributes can be analyzed in subjects having difficulties in their walking due to disease or injury. Employment of engineering principles and techniques has allowed for the measurement of human gait, permitting the assignment of values to these attributes. The combination of the engineering and health science has opened a relatively new field of science, ***biomechanics***, or mechanics of living systems, which strives to understand human musculoskeletal function, both in its normal and pathologic states.

Studying walking is one of major prerequisites for researchers trying to produce walking in completely paralyzed people or to improve walking in partially paralyzed people. Human walking has been studied and described more than any other movement, and scores of laboratories are dedicated to the assessment of walking, both normal and pathological. J. Perry wrote on the role of gait analysis in assessment of the pathological gait (1992):

> "...Identification of such patients' dysfunction requires an ability to recognize the subtle as well as obvious events and the knowledge of how to interpret the observations. The most convenient sensor is the trained eye of the practicing clinician. This permits assessment of the problem at any time and in any environment. Assessment of more complex situations, however, necessitates laboratory measurements. They add greater precision, provide information that cannot be obtained by eye and facilitate correlation of multiple factors..."

Under the general discipline of biomechanics there are two approaches in studying walking: *kinematics* and *kinetics* (Braune and Fisher, 1987; Chao and Cahalan, 1990).

Kinematics is the study of *rigid body* motion from a purely geometric point of view (e.g., displacement, velocity), without concern for the cause of the motion. Kinematics does not deal with the masses and moments of inertia of the bodies in movement or with forces exerted. A *rigid body* is a system of particles for which the mutual distances between all particles remain constant. Regarding biomechanics, parts of the anatomy may be considered rigid bodies, such as the lower leg, thigh, and pelvis.

Kinetics is the study of the relationship between rigid body motion and the forces and torques causing the motion. Force is caused by muscle contraction, gravitational attraction, and other physical and mechanical effects. Two basic types of forces and torques are described in the biomechanics literature: external and internal. External forces result from direct contact of one body to another (e.g., the foot striking the ground) and from gravitational forces. Internal forces

may be applied forces by the muscles and tendons or by constraint forces occurring on the joint contact surface, in the ligamentous structure and within the bone or between bone and prosthetic components.

The most commonly measured output variables are simple temporal and length measures. A wide variety of systems, reviewed by Winter (1991), have been developed to measure the simple stride measures: displacement, velocity and acceleration. Everything from simple foot-switches to instrumented walkways record temporal events, such as heel contact, foot flat, toe off, stance time, double support, etc. (see Figure 2.3.1). Instrumented walkways also yield step and stride lengths. Imaging techniques (cinematography, television and opto-electronics) not only yield the desired kinematic measures but also give us most of the necessary temporal and stride measures. Multiple exposure techniques, cinematography and special film digitizing systems have evolved to allow body coordinates to be extracted for computer analysis. Similarly, television has been used to advantage, not only as an instant playback, but also as input to special interface electronics for conversion to computers (Ferrigno and Pedotti, 1985). Also, very specialized opto-electronics and magnetic systems which require active optical or magnetic sources as markers are now commercially available.

Angular measurements of segments are not very often calculated or reported. Joint angles are relative and therefore tell us nothing about the absolute angle of each of the adjacent segments in space. In spite of this, in normal walking, the trunk can be considered to be almost vertical (actually it is biased slightly forward of vertical), thus the hip angle can be used to give good approximation of the thigh in space, and the knee angle, in turn, could yield the leg angle and the ankle angle could give a reasonable estimate of the foot angle in space. The definition of the three major joint angles of the lower extremities is presented in Figure 2.3.2.

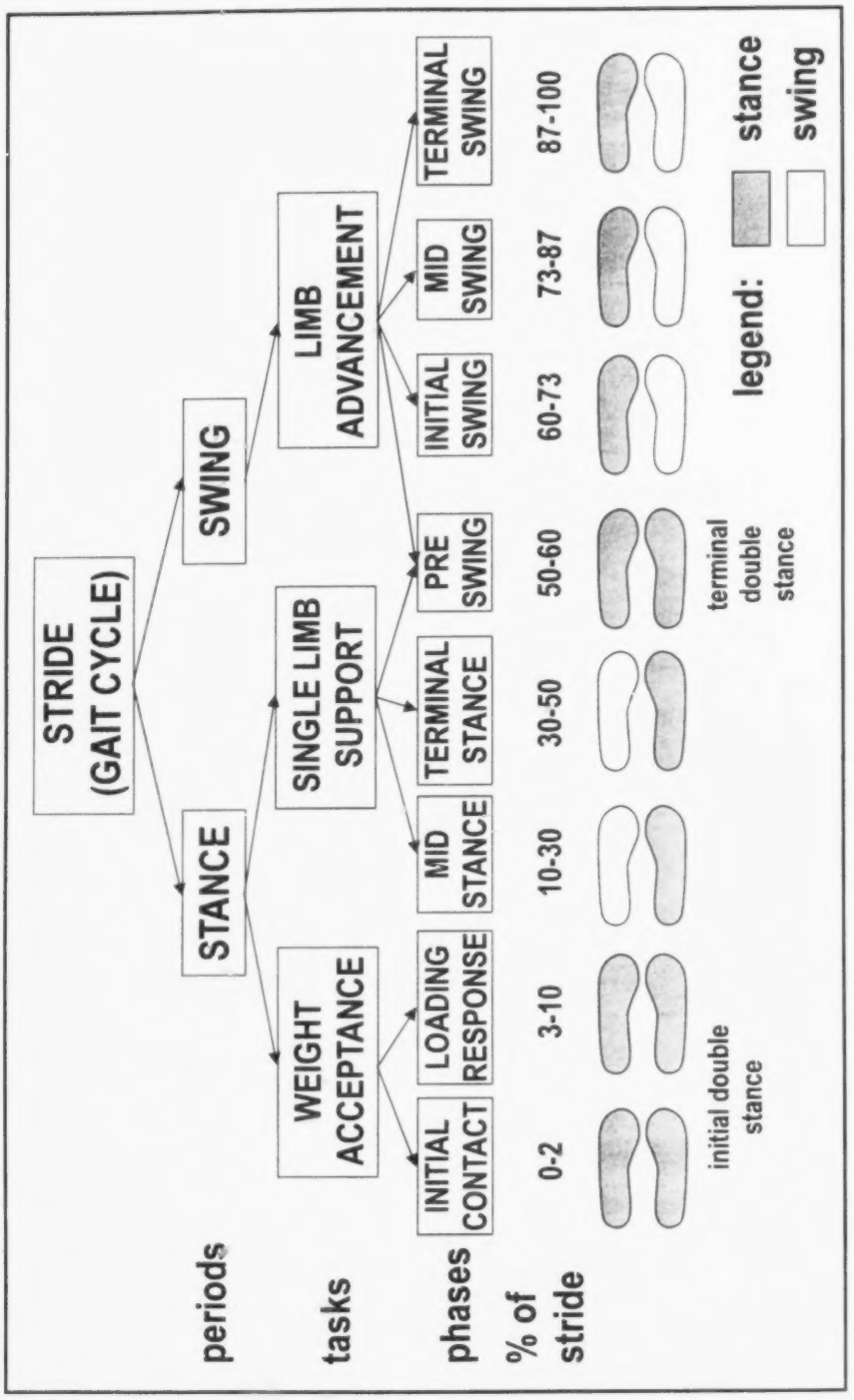

Figure 2.3.1. Temporal events in a gait cycle (modified from Perry, 1992).

Joint angles are measured with specially designed goniometers. The most often used principles employed in these devices are: rotation of the potentiometer; loading of the flexible strain gauge; and change in resistance by elongation of an elastic tube filled with conductive material (e.g., mercury). A single goniometer can make measurements only in one axis of the joint, but two or three may be mounted in different planes to make multi-axial measurements. Concern has been expressed about the accuracy of measurement provided by these devices, since they are subject to a number of possible types of error because the devices are fixed to soft tissue, not to the bone. The output of the device gives a relative, rather than absolute angle, and it may be difficult to decide what limb angle should be taken as 'zero', particularly in the presence of a deformity. In addition, the goniometric technique, although widely used, does not yield stride measures. In a similar way, *accelerometric techniques* provide limited kinematic measurements, but also do not yield stride measurements. *Inclinometers* are another similar devices which are used frequently to measure a body segment's angle relative to the ground force vector.

The *force platform (or forceplate)* is an instrument commonly used in gait analysis. It gives the total force applied by the foot to the ground, although it does not show the distribution of this force across the sole of the foot. Some force platforms give only one component of the force (usually vertical), but most give a full three-dimensional description of the average ground reaction force vector. The electrical output signals may be processed to produce three components of force (vertical, lateral and fore-aft), the two coordinates of the center of pressure, and the moments about the vertical axis. The *center of pressure* is the point on the ground through which a single resultant force appears to act, although in reality the total force is made up of innumerable small force vectors, spread over a finite area on the surface of the platform (force field). Obviously, although it gives very useful measures, a force platform is a piece of equipment which in its present form will never leave the laboratory environment. Another limitation in using the force platforms for analysis of walking of paralyzed people is the use of hand supports, such as parallel bars, wheeled walkers and crutches.

Metabolic energy consumption during walking would be a very important measurement in studying walking if the available measuring methods were more pleasant for the subject and if the measurement results were more explicit and easier to standardize. Measurements of oxygen uptake is an indirect method of measuring the energy consumption (i.e. metabolic cost) of different activities. It requires special equipment, the subject has to wear a face mask or special mouthpiece with a clip which closes the subject's nose and leaves only mouth to breath through the mouthpiece. Another problem with this method is the lack of a suitable baseline for energy consumption measurements in humans (Whittle, 1991). The energy requirements of walking can be expressed in two ways: the energy used per unit of time, or the energy used per unit of distance. The equipment required for these measurements is still too cumbersome to be used outside of the laboratory.

Velocity is a simple measurement that can be done by timing a subject while he or she walks a known distance. Other general gait parameters that can be obtained by simple measurements are the cadence and the stride length. Both can be determined either during the walking session or by using the video recording of the subject walking between two landmarks whose positions are known.

If more precise time measurements are required, exact timing of the gait cycle can be obtained in two different ways, using *footswitches* or *instrumented walkways*. *Footswitches* are more flexible for use inside and outside laboratory. Just two switches installed under each foot, one beneath the heel and one beneath the forefoot provide precise timing of heel contact, foot flat, heel off and toe off and the duration of the stance and swing phases. Data from two or more strides make it possible to calculate cadence and swing phase duration. Single and double support times can also be measured. Since the information provided by switches is very simple (ON/OFF), the signal from switches can be transmitted to the recording system through an inexpensive radio connection, which eliminates the limitations imposed by cables connecting the measuring device to the recording equipment. Also, small portable devices which measure timing of the walking are

already in commercial use. *Instrumented walkways* are used to measure the timing of foot contact and the position of the foot on the ground. Many different designs have been developed, usually individually built for a single laboratory. In a typical design, the walkway is covered with an electrically conductive substance such as sheet metal, metal mesh or conductive rubber. Suitably positioned, electrical contacts on the subject's shoes complete the electrical circuit. The conductive walkway is thus a slightly different method of implementing footswitches, and provides essentially the same information. Obviously, this measurement system is limited to laboratory use.

Pressure beneath the foot is a specialized form of gait analysis that may be of particular value in abnormal conditions, such as diabetic neuropathy and rheumatoid arthritis, in which the pressure may be excessive. Lord (1981) and Lord et al. (1986) provided a nice review of a number of systems which have been devised in an attempt to identify high pressure areas. Most of the foot pressure measurement systems are floor-mounted. It is more relevant, but also more difficult, to measure the pressure beneath the foot inside the shoe. Lord et al. (1986) and Webster (1992) pointed out that it is very important to distinguish between force and pressure. Some of the measurement systems measure the force (or 'load') over a known area, from which the mean pressure over that area can be calculated. However, this mean pressure may be very different from the peak pressure within the area, if high pressure gradients are present, which are often caused by subcutaneous bony prominences such as the metatarsal heads. *In-shoe devices* represent an alternative to floor-mounted measurement systems. The main difficulties with this type of measurement are the curvature of the surface, a lack of space for the transducers and the need to run large number of wires from inside the shoe to the recording equipment. For these reasons most systems of this type measure pressure only in selected areas, in contrast to the floor-mounted systems, which measure it over the whole area beneath the foot. This may not be disadvantageous if the purpose of the measurement is the detection of events during gait cycle. In-shoe devices with reasonably small number of sensory cells could provide valuable information for control purposes.

Electromyography (EMG) is the measurement of the electrical activity of a contracting muscle. Since it is a measure of electrical and not mechanical activity, the EMG cannot be used to distinguish between concentric, isometric and eccentric contractions, and the relationship between EMG activity and the force of contraction is far from straightforward. However, EMG combined with other kinematic and kinetic measurements could provide a valuable insight in muscle activity during walking. Recording of EMG can be done outside a laboratory environment which makes it a good candidate as a signal source for control purposes in walking of paralyzed people. EMG measurements in normal walking (Perry, 1992) have been used as a basis for stimulation patterns for FES-assisted walking (Kobetic and Marsolais, 1994, Handa et al. 1987).

2.3.2 Force sensors

If the sensors were to substitute for cutaneous receptors, then they should measure pressure and not force, because that is exactly what cutaneous receptors measure (Webster, 1992). Force sensors used under relatively large areas under the foot are sometimes refered to as pressure sensors. This is not correct in the case of large force gradients, such as those under metatarsal areas, because measurement of pressure may be imprecise.

After testing some of the commercially available force sensors, the choice was made to continue design with Interlink Electronics force sensing resistors (FSRs) which satisfied most of the criteria listed in previous section and presented only resolvable problems (Interlink Electronics, 1990a). The FSR is an easy-to-use, readily available, relatively inexpensive, highly durable sensor with highly reproducible measurements. It converts contact force into an electronically readable format. When actuated with increasing force, the FSR responds with a drop in resistance. Actually, the FSR lies somewhere between a force and pressure transducer. A typical FSR will show a resistance that varies roughly as the reciprocal of the square root of the area of the applied force, which is characteristic of a pressure sensor. This holds true under the condition where the force footprint is smaller than active area of the FSR. They can be used as a pressure sensor when the

area of the applied force is large compared to the active area of the FSR. The same sensor can be used as a force sensor in different mechanical arrangement, where the force footprint can be held constant in area and position. Since none of the conditions is satisfied exactly in measuring the contact between the foot and the shoe, the FSR must be used only as a qualitative sensor.

The construction of a typical FSR is shown in Figure 2.3.2. The force sensing part of the sensor is based on two polymer sheets. A conductive pattern is deposited on one polymer in the form of a set of interdigiting electrodes. The electrode pattern is typically on the order of 0.4 mm finger width and spacing. On the other polymer sheet semiconductive polymer is deposited. The two sheets are faced together so that the conductive fingers are shunted by the conductive polymer. When no force is applied to the sandwich, the resistance between the interdigiting electrodes is quite high, usually 1MΩ or more. With increasing force, the resistance drops, following an approximate power law over a limited range.

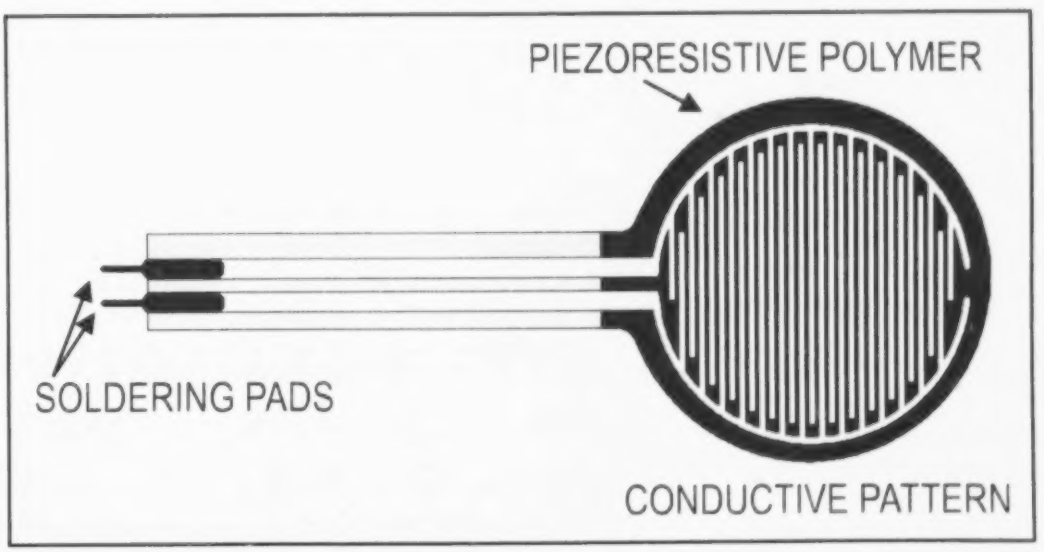

Figure 2.3.2. The construction of force sensing resistor (FSR).

The site of the major problem recognized with that sensor was the packaging, which originally did not provide enough protection and mechanical support for the sensor and the contacts. The failures most often seen were at the connection where the wires were soldered to solder pads which are crimped through the conductive silver strips (see Figure 2.3.2).

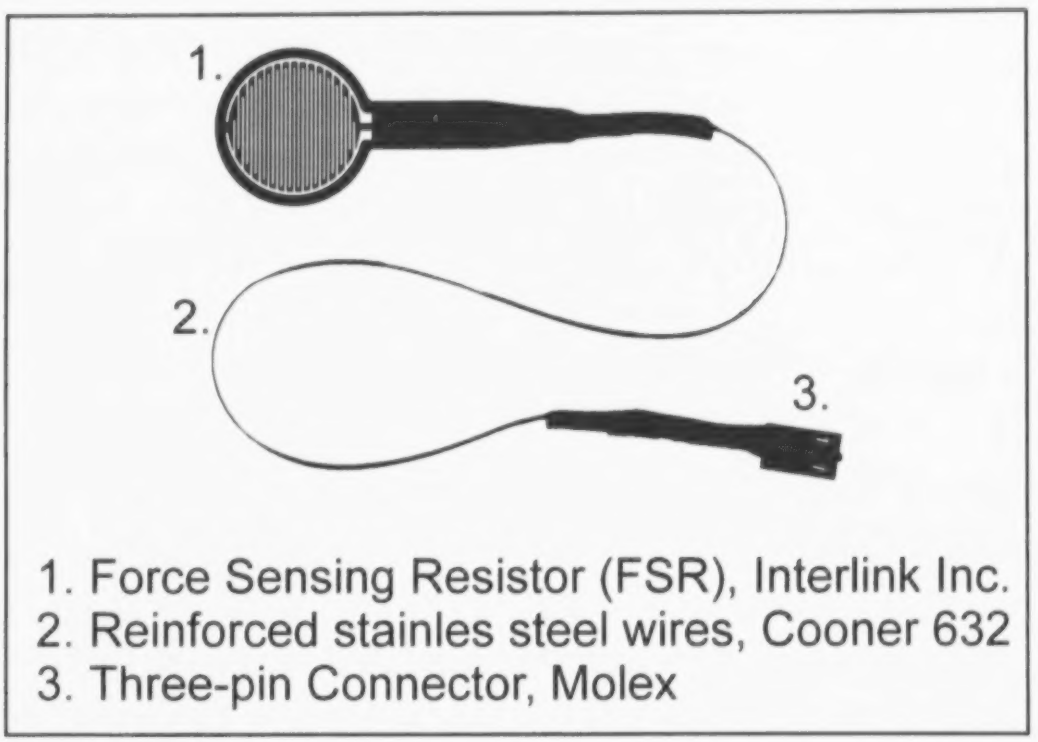

Figure 2.3.3. Force sensor and its connections.

Inadequate wire (too thick or too rigid) which transferred the stress to the contact pad would usually losen the connection between the pad and the silver strip, and the sensor will fail. One way to resolve the problem was to choose very flexible and durable wire, to eliminate the solder pads and to protect the wires from stress. Cooner AS 632 is a stainless steel wire known for its durability in design of intramuscular and nerve electrodes for chronic animal recordings and stimulation. Instead of soldering the wires to the contact pads, they were connected to the silver

strips using a two component conductive epoxy 2400 made by Circuit Works. To provide strain relief for the wires a surgical thread was attached close to the soldering pads and to the connector at the end of a 30 cm cable. The thread was 1.5 cm shorter than the wires, which provided good stress protection for the wires. To protect the wires even more, they were placed together with the surgical thread into a 1.5 mm heat-shrinkable tubing. The resulting sensor, as presented in Figure 2.3.3, was attached on the bottom of a shoe insole, usually in sets of four: one of each under the heel, big toe, medial metatarsal and lateral metatarsal joint. The use of these shoe insoles in a portable recording system for assessment of FES-assisted locomotion is reported in Tepavac et al. (1992) in which the part related to the shoe insoles instrumented with force sensors and a corresponding signal conditioning was contributed by the author of the present thesis.

The sensosr usually operated without failures in a subjects' shoes for several weeks up to several months. The problem with this design was related to the conductive epoxy which, after being exposed to the very unfriendly environment over time, changed its characteristics. The connection between the flexible wire and the silver strip loosened and the sensor failed. The repair was usually impossible because it was difficult to remove the old conductive epoxy without damaging the silver strips. Wire breakage was recorded only once in more than 20 sensors manufactured which indicated that the wire and the thread reinforcement worked better than the wires used in previous designs.

One more problem was continuously present: the connector at the end of the 30 cm cable coming from the sensor did not provide a reliable connection. This connector was used because the long wire from the ankle of the instrumented foot would be too expensive if Coonner 632 was used for the purpose of connecting the sensors and the signal conditioning device. In addition, it would require as many long wires as there are sensors if the connector was not used. The solution of this problem was found with 'Tinsel' wire (Lazorthes 1985): a lead made of seven wires, each of them being two stainless steel ribbons wrapped around a polyester arbor which increases mechanical resistance, and therefore ensures a longer lifetime. Basically, the surgical thread used

outside the wires in the previous design was replaced with the polyester arbor inside the wires in this design and simplified the design. In addition, since Tinsel wire is cheaper than Cooner AS 632, the connector between the wire coming from the sensor and the cable leading to the signal conditioning device was eliminated.

The next step in the design was to reduce bending on the soldering pads by providing better mechanical support. A decision was made to produce customized shoe insoles in different sizes and with a different number of FSRs. The use of similar insoles instrumented with the same sensors for laboratory testing was reported by Andrews et al. (1987, 1988), Zhu et al. (1990), Wertsch J.J. et al. (1992), Tepavac et al. (1992), and Ferencz et al. (1993). Two materials were considered: TEPP2 and RHENOFLEX ,provided by a local orthopedic company. Both materials are thermo-plastic and are used in various orthopedic designs. TEPP2 is a more flexible material, having fabric on one side and thermo-sensitive rosin exposed on the other. The side covered with fabric is very suitable for the upper side of the shoe insole. The base of the insole was produced using the same TEPP2 material with the fabric side at the bottom, or using RHENOFLEX, which is a more rigid material and has both sides covered with fabric (see Figure 2.3.4).

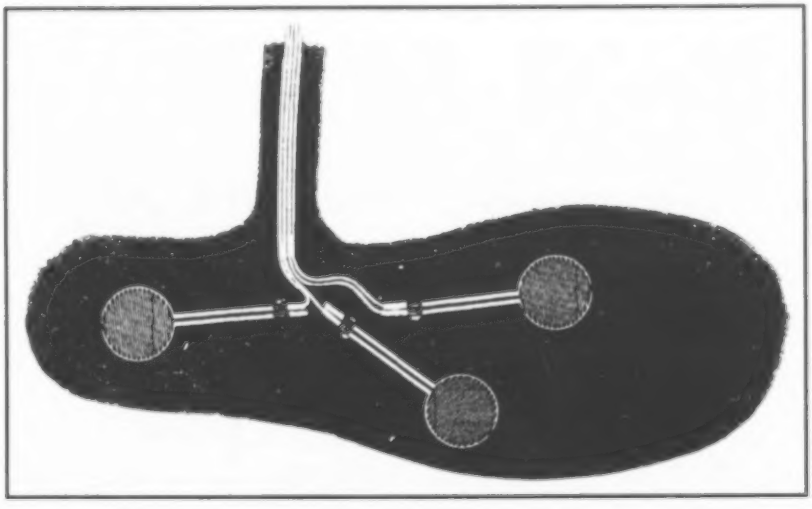

Figure 2.3.4. Shoe insole instrumented with force sensing resistors.

Both designs worked well under laboratory conditions. The remaining question was: how long will this insole operate in an unfriendly environment of a shoe. One two-sensor insole with both sides made of TEPP2 has been in everyday use by the subject for more than 16 months (6 hours/day on average) without failure. Two insoles made in the same way with three sensors each have been used in the laboratory for more than 15 months (5 hours/week on average) without failure. It is expected that insoles with the more rigid RHENOFLEX material in the bottom would last even longer than those made only of TEPP2 material due to better mechanical support. Two insoles produced with the RHENOFLEX material in the bottom have been in laboratory use for more than a year (2 hours/week on average) without failure. To restrain forces on the wires due to the bending of the foot during walking, an extended tongue of the TEPP2 material and the shrinkable tubing protects the exiting wires up to a level above the shoe.

The new insole design exceeded all expectations. It showed that packaging is the major problem with FSRs and, if it is resolved properly, they can be a reliable source of sensory feedback for control purposes.

To enter the competition and replace the FSRs as the most often used force sensors, new force sensors have to beat FSRs in availability and price. A better quality of response with smaller hysteresis and more linear characteristics is already achievable (Beebe 1994).

2.3.3 Goniometers

Goniometers measuring the joint angles were used in several experiments together with foot switches or force sensors. The flexible strain gauge goniometer M180 (see Figure 2.3.5), manufactured by Penny and Giles Biometrics Ltd. (Penny and Giles, 1994), was used to measure hip abduction-adduction, hip flexion-extension, knee flexion-extension, ankle inversion-eversion and ankle dorsal-plantar flexion of the disabled leg (Figure 2.3.6.) The working mechanism of this goniometer is as follows: Between the two endblocks inside the protective spring, there is a

composite wire which has a series of strain gauges mounted around the circumference. As the angle between the two ends changes, the change in strain along the length of the wire is measured and this is equated to angle. The design is such that only angular displacements are measured. If the two ends move linearly relative to each other, within the limits of the telescopic endblock, without changing the relative angles between them, then the outputs remain constant, i.e. they need not be positioned over axis of rotation. To measure motion in more than one axis, which was required for hip and ankle joints, both outputs from the two-axis goniometer were used. Goniometers were attached to the subject's skin by adhesive tape, which held them in place during the experiment. They are considered only as being part of the laboratory setup because of their high sensitivity to mechanical stress and high price.

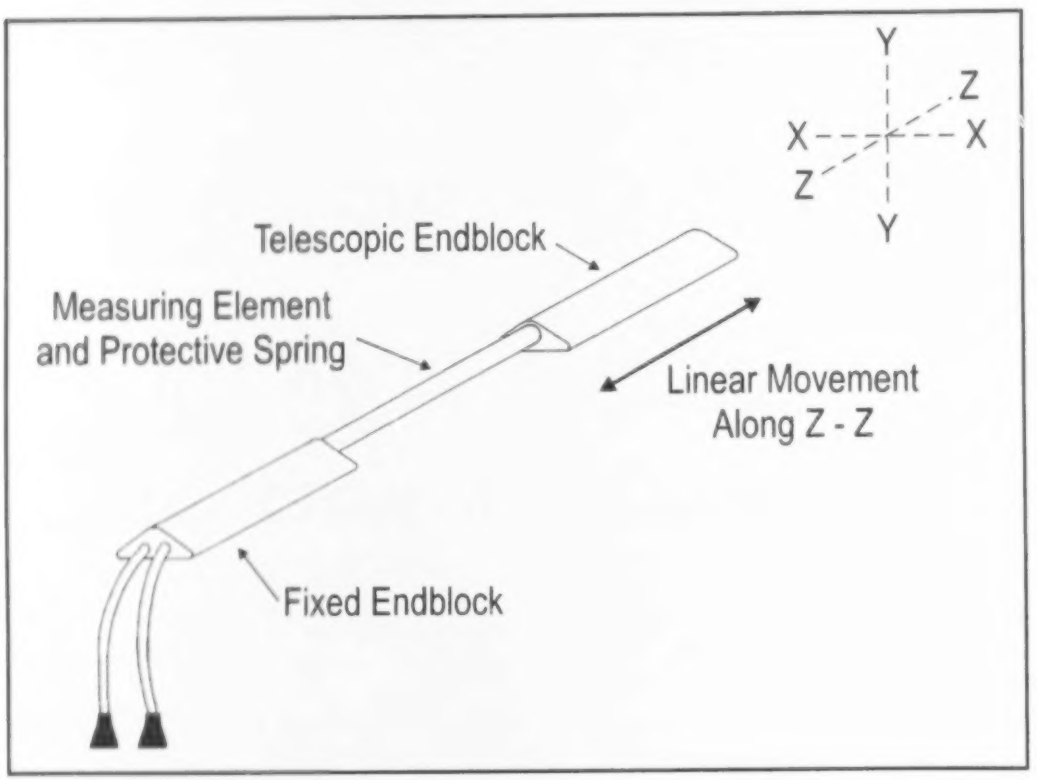

Figure 2.3.5. Twin axis goniometer M180. This goniometer measures the relative position of the two end blocks in the X and Y planes.

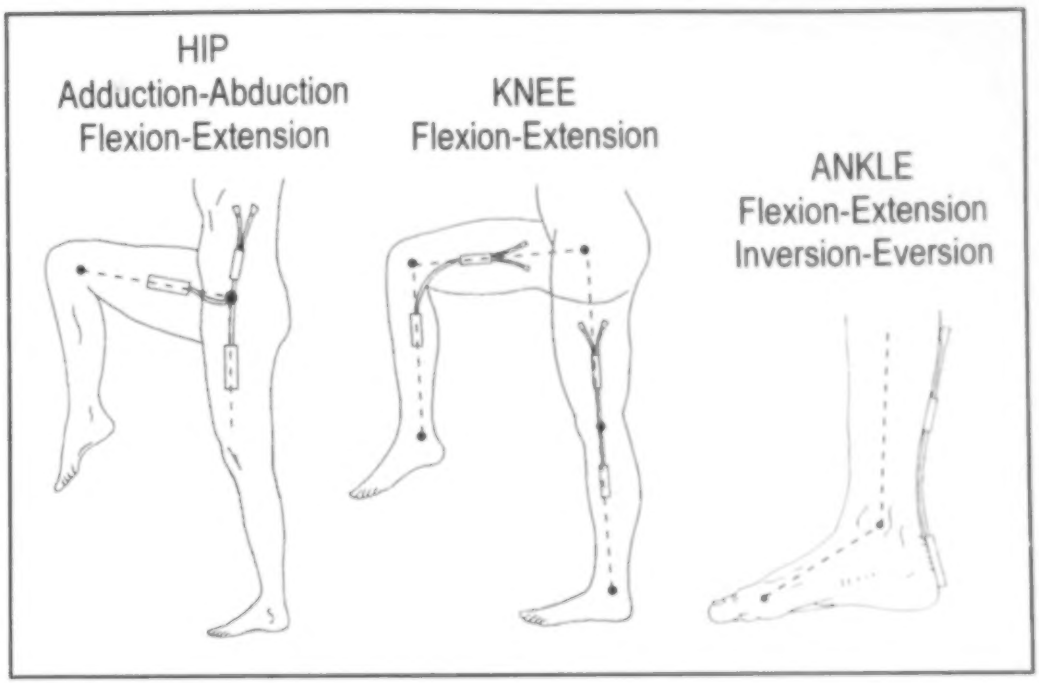

Figure 2.3.6. Twin axis goniometer M180 installed across: A) hip joint to measure its abduction-adduction and flexion-extension angles; B) knee joint to measure its flexion-extension angle; and C) ankle to measure its flexion-extension and inversion-eversion angle variations.

2.3.4 Inclinometers

Inclinometer is a device measuring a relative angle between a body segment and the ground force vector. In the experiment presented in Section 3.4 we used four inclinometers (Midori UV-1B) installed on subject's braces to measure inclination of her hips in two orthogonal planes, ie. hip flexion-extension and adduction-abduction.

2.4 SIGNAL CONDITIONING HARDWARE

Force sensors are highly nonlinear devices, whose resistance vs. force partially follows a power law, and as such are much more suitable for small force measurement than for the range expected inside the shoes, under the subject's feet.

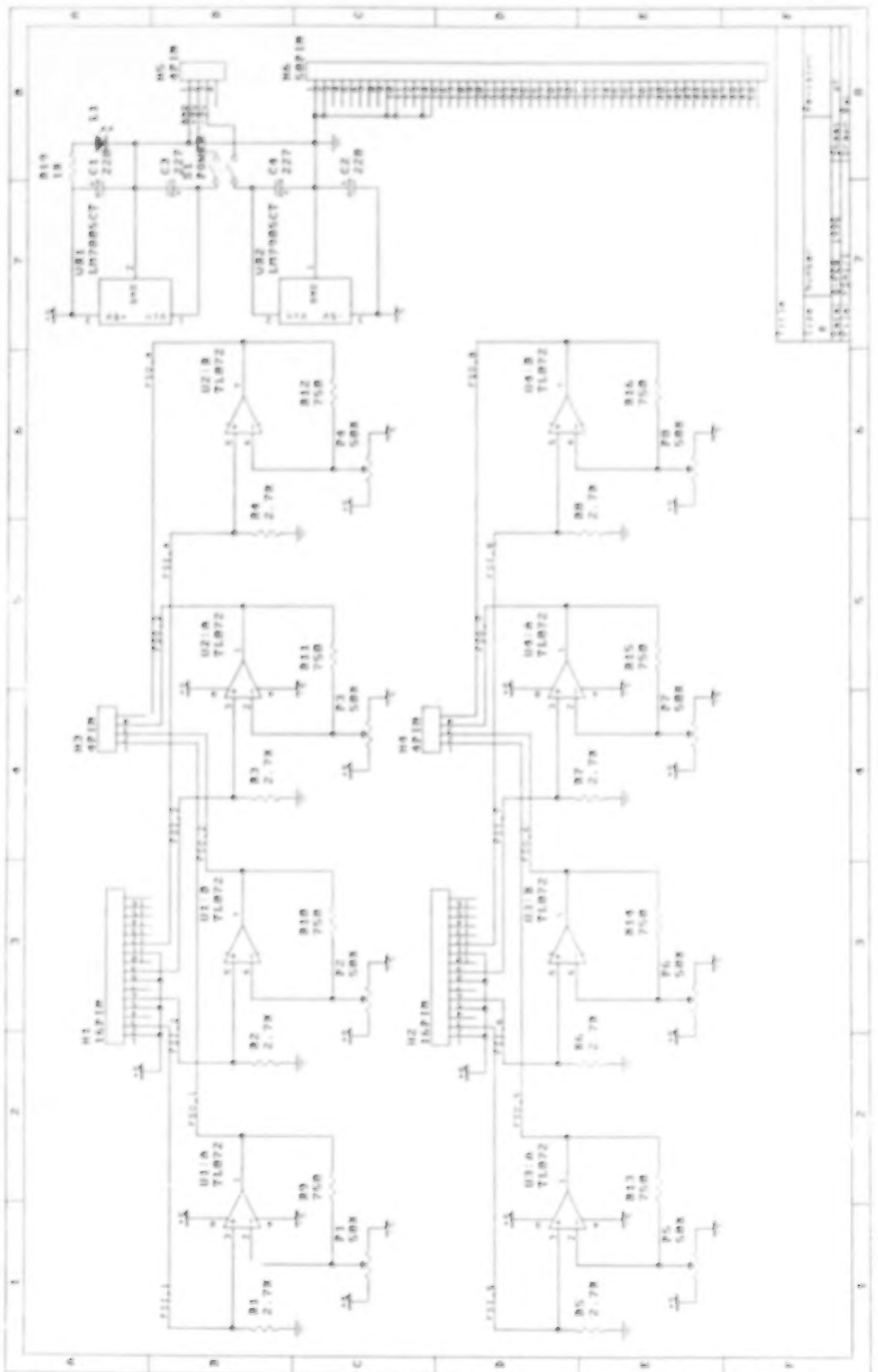

Figure 2.4.1. Schematics of the signal conditioning electronics for force sensors. A 50-pin connector provided mechanical coupling with the rest of hardware interface presented in Figure 3.6.2.

A method of virtually reducing the sensitivity of the FSRs in the low-force range, proposed by Interlink (1990b) was used in designing signal conditioning electronics. That eliminated the saturation of the output signals at low forces and expanded the measurable range of force.

Signal conditioning electronics for eight force sensors was designed in the form of printed circuit board with two layers. The voltage regulator provides a constant voltage supply to the sensors. A one stage low-noise amplifier (JFET, TL072) is used for both offset and gain regulation. In the current device the gain is fixed and the offset is regulated by potentiometers P1 - P8 for channels 1 through 8 respectively. The schematics of the electronic circuit is presented in Figure 2.4.1.

Goniometers: Signal conditioning electronics for goniometers consists of two stages. Preamplifiers for six joint angle measurements already existed in the laboratory and their design followed the manufacturer's specifications (Penny and Giles Biometrics Limited, 1994). General purpose laboratory instrumentation amplifiers were used in the second stage of amplification.

2.5 DATA ACQUISITION AND I/O SYSTEM

Two different data acquisition systems were used, AXOTAPE, made by Axon Instruments, Inc., for off-line evaluation of learning algorithms and AT-MIO 16DH, made by National Instruments, Inc., for real-time input/output application of the control algorithm based on machine learning. Both systems are described in detail in Appendix A.

2.6 DIGITAL SIGNAL PROCESSING

Digital filtering was used to reject high frequency noise. A low-pass, phase cancelled, fourth-order Butterworth digital filter (Barr and Chan, 1986) was designed in the form of a computer program HEX_FILT.EXE, written in the C programming language and used to filter data acquired and stored in the Axotape FETCHEX 5.2 data file format. More details about the program can be

found in Appendix C and the program is included in the computer disk which is part of this thesis. Decision on the cut-off frequency was made following the work by Winter et al. (1974) and Winter (1991) who suggested setting the cut-off frequency at the 6th harmonic of the stride frequency. The exact cut-off frequency can be decided after a residual analysis as detailed in Winter (1990).

The cut-off frequency was decided using the fastest stride frequency for the group of subjects who participated in this study. The fastest strides recorded and used in this study are presented in the Table 2.6.1. Based on these values, 2.5 Hz was decided for the cut-off frequency. This frequency satisfied Winter's criterion for most of the signals used and also provided a chance to reduce the sampling rate further in order to reduce the training sets for machine learning techniques. Small training sets were important in the beginning of this study due to relatively slow computers that were used for machine learning (IBM PC compatible 386 SX/25 MHz). In more recent phases of this study, a faster computer was used (IBM PC compatible 486 DX/50 MHz), which allowed for a much larger size of the training set and eased the restrictions on the sampling and low-pass filtering frequency. The exact filtering and sampling frequencies are given with each of the experiments.

Table 2.6.1. The fastest strides measured in data recorded from subjects participating in this study and the corresponding cut-off frequencies.

Subject	C.C.	C.H.	N.D.	L.S.	M.W.	L.W.
shortest stride (s)	1.8	1.76	2.92	1.88	3.16	3.78
low-pass cutoff	3.33	3.41	2.05	3.19	1.90	1.59

Calculation of derivatives (velocities) from the low-pass filtered data was achieved through a finite difference. The only necessary precaution was that the calculation must be done over two sample intervals in order that the velocity can be defined at each sample time. For example, the velocity at the i th interval of time, $V_i = (X_{i+1} - X_{i-1}) / 2T$, where X_{i+1} and X_{i-1} are the

coordinates at the $(i+1)$th and $(i-1)$th interval respectively, and T is the sampling interval. Similarly, accelerations can be calculated as $A_i = (V_{i+1} - V_{i-1}) / 2T$.

2.7 MACHINE LEARNING TECHNIQUES

Implementation of machine learning techniques (MLTs) to problems in control of biomedical systems can be divided into several steps, as illustrated in Figure 2.7.1. In general, there is a sequence of events which occur before the actual MLT involvement, and also, there is another sequence of MLT-related events. Before signals are presented to the MLT learning algorithm, they have to be acquired, preprocessed, quantized and encoded (if required). Data acquisition and storage can be done using commercially available software (e.g., Axotape, Axon Instruments, Inc.), or by custom programs from within integrated control system, such as the program FESCONT described in the Section 2.8.3. After signals are stored, various signal processing techniques are used to enhance signal-to-noise ratio, modify and increase the amount of signal-based information. Data prepared in this way are then quantized and encoded, if required by a particular MLT, which makes them ready for use with MLTs.

Implementation of MLT-based control involves the following stages. The first phase is an ***MLT TRAINING***. Data prepared according to the procedure described above are presented to the MLT learning algorithm (see Appendix B). MLT maps the set of input data (representing a domain of the mapping) into the set of outputs (representing a codomain of the mapping). The actual training is monitored and it continues until the performance of current MLT trees reaches training criteria. Training criteria are usually preset in form of an allowed percentage of wrongly mapped bits in signal samples (data examples) in the training data set. The difference between actual and synthesized output is expressed as a percentage of wrongly restored samples in the training set and it is called ***training error***. The next phase is an ***MLT TEST*** during which trained MLT decision trees are presented with a new set of input data and expected to produce (predict) the appropriate

output. This test can be done either off-line (***Off-line TEST***) or during new walking round (***Walking TEST***). The predicted output is then compared with known data. The difference between actual and predicted outputs is expressed as a percentage of wrong predictions (in the test set) and it is called ***test error***. If the test error satisfies preset criteria, the trained decision trees are ready to be tested in a real-time control application (***Walking CONTROL***). Depending on the functional task, performance of the whole MLT-based control system is assessed by measuring appropriate functional error. If the functional error is too high, new decision trees can be trained using the same training data set or new data can be prepared from the same signals by varying signal preprocessing parameters. The success of this process is based on the researcher's intuition and experience. Some attempts to formalize the path toward a good functional results are presented in this thesis.

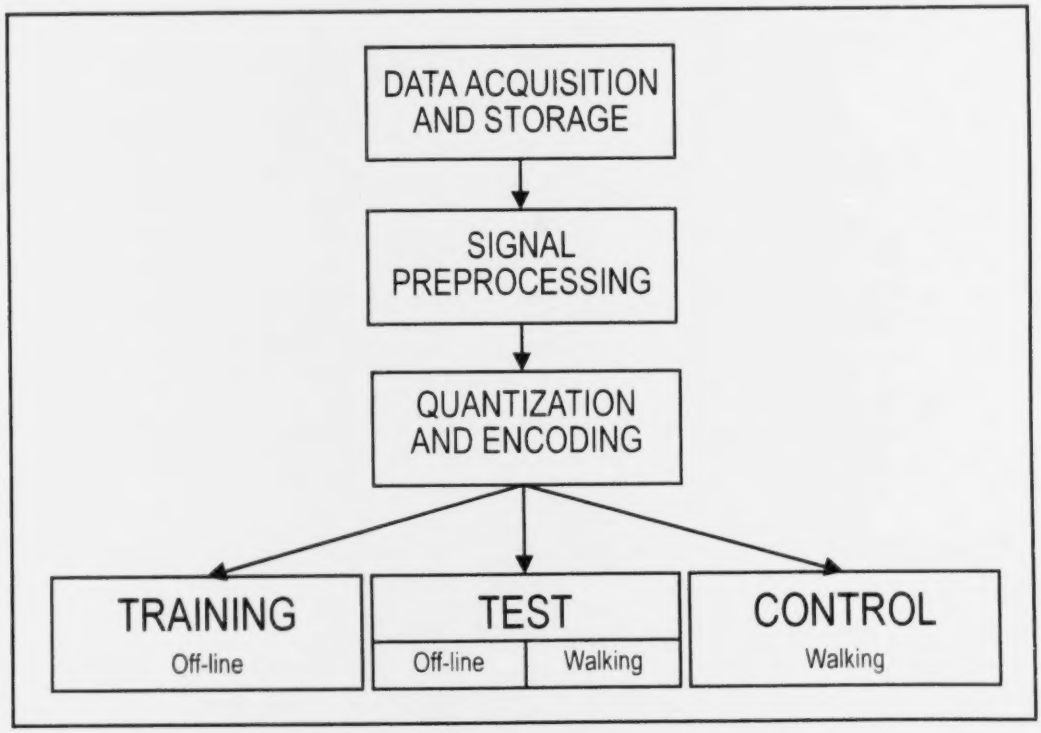

Figure 2.7.1. Data preparation for use with MLTs, and the sequence of MLT implementation phases in control applications.

Two machine learning techniques were evaluated for automatic design of a rule-based control of FES-assisted locomotion of spinal cord injured humans. The task was to learn the invariant characteristics of the relationship between sensory information and the FES-control signal(s) by using off-line supervised training. Sensory signals were recorded either from various traditional sensors including force sensors, goniometers, and inclinometers, or from natural biological sources such as mixed nerves in the cat's hind limb. Force sensors were installed in insoles of subject's shoes, goniometers were attached across the joints of the affected leg, in the case of subjects with incomplete spinal cord injury, or across the joints of both legs, in the case of the subject with complete spinal cord injury. Inclinometers were installed on subject's braces to measure inclination of her hips in two orthogonal planes, i.e. hip flexion-extension and adduction-abduction The FES-control signal(s) consisted of pulses corresponding to time intervals when the subject pressed on the manual push-button(s) to deliver the stimulation during FES-assisted ambulation. When biological sensory feedback signals were used, the FES-control signals were:

- binary pulses representing time intervals when muscles should be contracted, or
- a digitized analog signal approximating the corresponding EMG signals.

The machine learning techniques evaluated were adaptive logic networks (ALNs) and inductive learning algorithm (IL).

2.7.1 Adaptive logic networks

The adaptive logic network (ALN) is a type of artificial neural network for supervised learning. It can be considered a special type of the feedforward multilayer perceptron in which the signals in the network are restricted to be boolean (binary) after a layer of processing units that act on whatever other types of signals are present (reals, integers, etc.) to produce boolean values. The result of machine learning process, is one or more ALN binary trees with a basic structure illustrated in Figure 2.7.2, where the units that convert signals to booleans are not shown. At the level of connections of the tree to those units, there may be several connections to the same

boolean signal, so the term 'tree' is only appropriate for the upper portion of the structure, above the level of inputs.

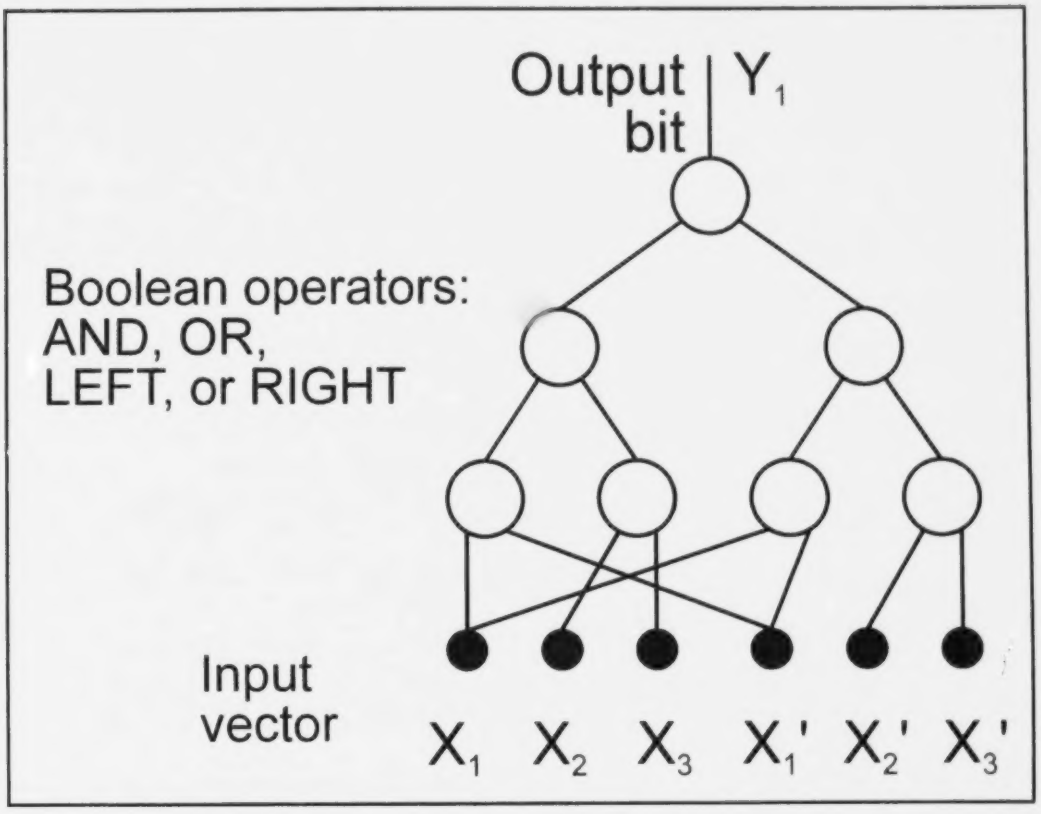

Figure 2.7.2 Seven node ALN with three input variables. To enable synthesis of monotonic decreasing functions, complements of the input bits are also present in the input vector (primed).

Two versions of ALN were used during the course of this thesis project: the one that incorporated Atree versions 2.7 and earlier will be denoted in further text as ALN V.2[4], and the one that incorporated Atree version 3.0 will be denoted as ALN V.3. ALN V.3 has adaptive linear threshold units (LTUs) that can act on reals, integers, etc., while ALN V.2 has fixed operators such as

---

[4] ALN V.2 Simulation Package which contains the program Atree version 2.7 is available from Department of Computing Science, University of Alberta, Edmonton, Attn. Dr. W.W. Armstrong or it can be accessed by ftp from: ftp.cs.ualberta.ca [129.128.4.241], file: /pub/atree/atre27.exe in binary mode.

threshold units to encode reals into booleans. In both cases, the adaptive parts of the structure form a tree.

A learning algorithm for the adaptive boolean logic element was introduced and patented by Armstrong (Armstrong (1971, 1976), Armstrong et al. (1972, 1990, 1991), Armstrong and Godbout (1974), Armstrong and Gecsei (1979)). A description of the class of learning algorithms and their hardware realization can be found in Appendix B, but the concepts are briefly introduced here. The nodes of the ALN tree are of two types: 1. adaptive elements, and 2. leaves. Each adaptive element is a two-input logic gate, which can be any one of the following four boolean functions: *AND*, *OR*, *LEFT*, or *RIGHT*, depending on its state. The last two functions, defined as *LEFT(a,b)=a* and *RIGHT(a,b)=b*, were introduced to cut out poorly performing subtrees. The state of an adaptive element is determined by two counters which change only during training. Leaves are the nodes of the first layer of an ALN tree used to connect binary inputs from the encoder to the tree.

The following characteristics qualify ALNs to be used for pattern recognition and synthesis (or mapping) of functions in real-time control processes:

- supervised learning algorithm (it learns from examples);
- low output sensitivity to input noise (Bochmann and Armstrong, 1974);
- lazy evaluation of logic functions (Armstrong et al., 1990), which results in an exponential speed-up of a simulated ALN (the larger the tree, the smaller the fraction of the tree that is likely to need evaluation for a given input vector);
- very fast execution of the learned function;
- easy hardware implementation (since all digital circuits are based on logic gates).

ALN V.2 deals with continuous quantities by encoding real numbers into bit strings. The boolean functions AND, OR, LEFT and RIGHT acting on booleans are monotonic increasing. A boolean *1* is considered greater than a boolean *0*. A monotonic increasing boolean function never decreases

its output when an input is increased. Atree of AND, OR, LEFT and RIGHT gates is appropriate for approximation of monotonic increasing functions. To be able to approximate monotonic decreasing functions or functions which are non-monotonic, the input vector is doubled in length to include complements of encoded values. After the input vector is formed, all bits are randomly connected to the first layer of the binary tree (leaves). The output of each tree is just one bit of the output vector, meaning that there must be as many binary trees as there are bits in the output vector. After the training is finished, ALN binary trees are used in predicting output events from the input information supplied. To obtain real results with ALN V.2, the output vector has to be decoded, which is the inverse process of the encoding. In ALN V.3 the network learns a realationship among variables, and a separate decision tree is used to compute the function derived from the relation.

The critical characteristic of an adaptive logic algorithm is that training time may become very long if the encoding and training parameters are not properly chosen. A long training time may result from large input and output vectors, large neural network trees, or an inability of the chosen structure to complete a particular learning task. Training was optimized with regard to the accessible features that can influence the critical characteristic of a particular algorithm (Kostov et al., 1992). The performance of the learning algorithm was tested by counting the number of incorrectly restored samples in a training set (training error) and incorrectly predicted samples in a test set (test error).

2.7.1.1 Encoding real numbers into binary vectors in ALN V.2

To process continuous quantities, such as analog signals, ALN V.2 requires that those numbers be quantized and converted into binary numbers. Before quantization and conversion, the amplitude range of each signal is decided, or calculated automatically by subtracting the minimum of the signal from its maximum. If the range is decided intuitively, it should cover all expected amplitude values throughout all phases of ALN V.2 implementation (e.g., training and test phase).

Amplitude quantization reduces an analog signal to one of a number of distinct levels so that the actual measurement can be digitally stored and processed in the minimal number of digits (bits) and still preserve physical characteristics of the measurement. This should not be confused with quantization occurring during signal acquisition process. Both data acquisition systems used during the course of this thesis project are designed to output 12-bit quantized digital measurements from analog signals. This number of bits accounts for $2^{12} = 4096$ discrete levels spread over the ±10 V of the input range. The precision provided by such a high number of bits is usually too high for low frequency, low noise signals such as those recorded during gait analysis. For this reason, the number of quantization levels is one of the parameters adjusted during the preparation of the analog signals before they are presented to the learning algorithm of ALN V.2. Two approaches to the quantization were explored: linear quantization with quantization levels equidistantly distributed over the whole amplitude range, and, so called, 'expert quantization', where the researcher distributes the quantization levels intuitively taking advantage of being able to group more encoding levels in those parts of the amplitude range where significant details in signal appear and spreading quantization levels apart, where the signal does not have many important details.

A digital representation of analog measurements is stored in the computer's memory in binary form. Since the ALN V.2 algorithm requires binary inputs and outputs, it could be concluded that these original binary numbers satisfy the requirements and that further encoding is not necessary. However, there is a serious problem with using a binary representation of integer numbers in non-perfect communications. The significance of all bits in a binary word is not the same (see Figure 2.7.3). The rightmost bit changes its value at every consecutive quantization level and it is known as the LSB (the least significant bit). On the other hand, the leftmost bit is known as MSB (the most significant bit) and it is equal to 0 for the first half of the quantization range, and equal to 1 for the second half. The other bits are located between LSB and MSB and their significance is greater than that for LSB, but smaller than that for MSB.Obviously, the importance of the information

carried by different bits in binary word is not the same and the error in LSB will not be equivalent to the error in MSB.

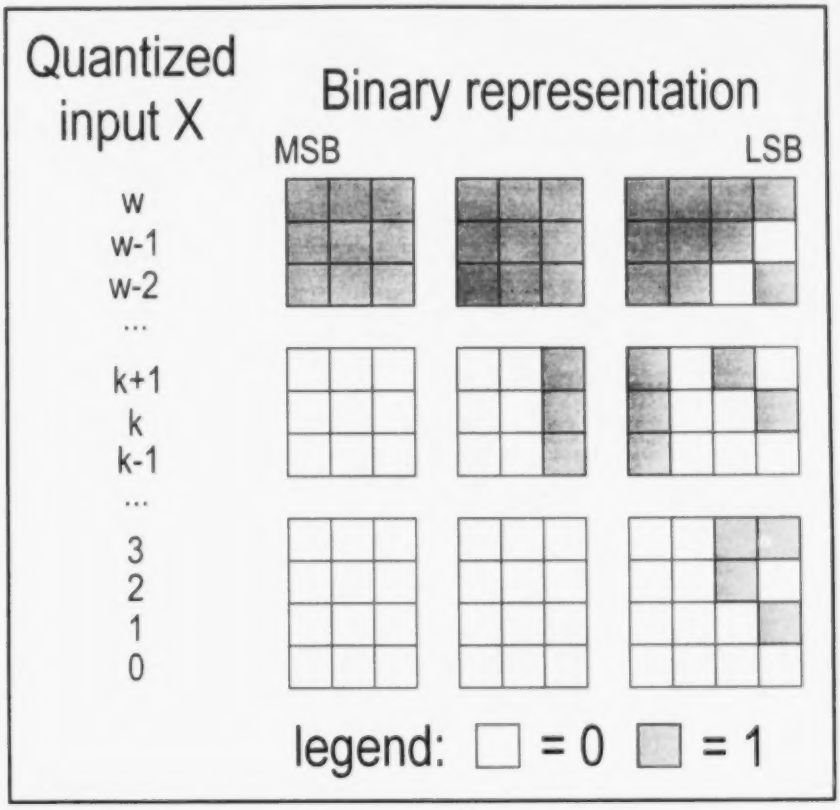

Figure 2.7.3. Binary representation of quantization levels.

After the amplitude range of each signal and appropriate number of quantization levels are decided, the next step in data preparation for adaptive training of ALN V.2 is ***encoding***. The only reason to use encoding is to put about equal weight on each bit of the representation. One technique, named ***random walk encoding***, has been described by Smith and Stanford (1990) and is illustrated in Figure 2.7.4. Given a sequence of *W* quantization levels of a real variable, *W* points in a Hamming space $\{0,1\}^n$ are generated as follows. A randomly produced binary vector is chosen to represent the first quantization level. For each subsequent integer, a specified

number of bits p, picked at random, are changed to create the next vector. This is repeated $w-1$ times until all w quantization levels are encoded into a unique binary vector. If the step $p>1$, the number of bits is usually larger than the minimal number required for a given number of quantization levels. The larger the step p is, the more balanced will be the significance of the bits in the binary vector. To decode the output binary vector, the search through the random walk is performed for the closest bit vector. This is taken as the true result. This technique was successfully used with adaptive logic networks during evaluation and feasibility studies for applications in the control field (Armstrong et al., 1990; Kostov et al., 1992).

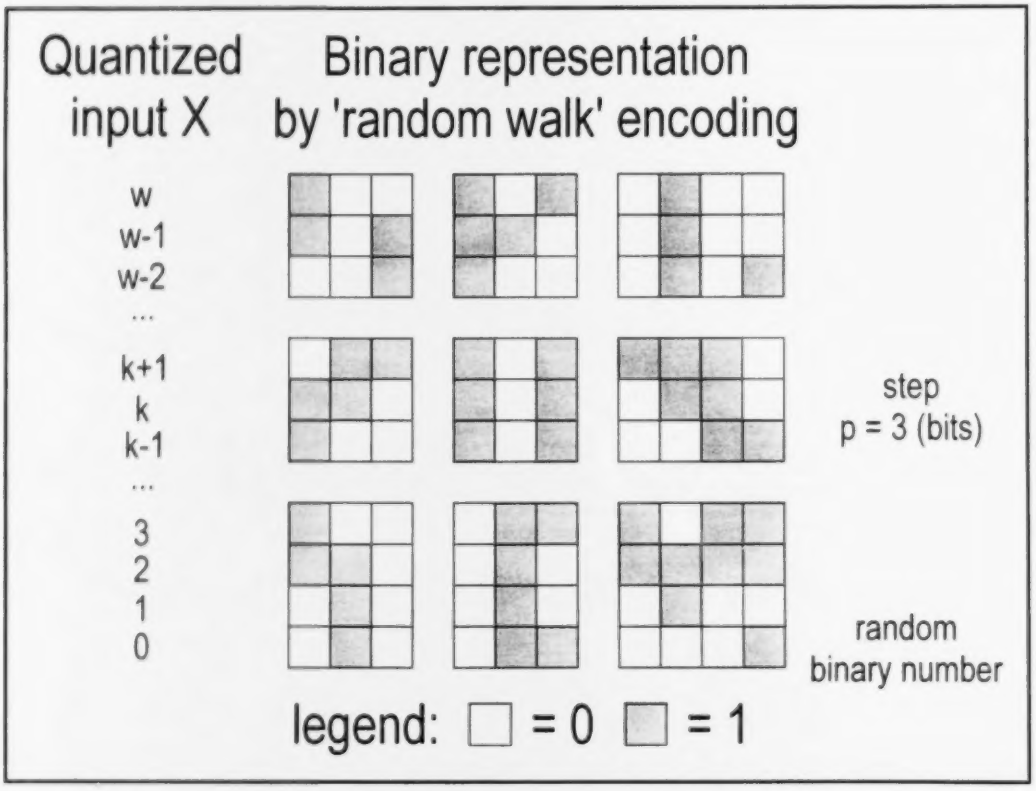

Figure 2.7.4. Random walk encoding technique.

Although the results with the random walk encoding technique were encouraging, the reproducibility of the results was not very satisfacory. Randomized encoding usually produced unpredictable results. Searching for a more linear encoding, the ***'thermometer' or unary encoding*** was tested resulting in more 'controllable' error (Kostov et al., 1993). Introduction of unary encoding was not an improvement of the ALN learning technique, but it provided a better way for the ALN V.2 to demonstrate its powerful adaptation and generalization features. Unary encoding is illustrated in Figure 2.7.5. This encoding technique is not optimized with regard to the number of bits used to represent particular number of quantization levels. In its initial form it used $w-1$ bits to represent w quantization levels. Practically, to encode the kth quantization level, all leftmost k bits in the binary word are set to 1 and the rest to 0. To decode the binary vector produced by ALN V.2 trees, all bits reading 1 are grouped to the left-hand side of the binary vector, leaving bits reading 0 at the right-hand side. This is equivalent to decoding by counting number of trees producing one. Unary encoding reduced the test error of ALN V.2 at least to half compared to the best results obtained by random walk encoding.

Although the large number of encoding bits used in unary encoding may appear as a significant burden on the computer's processor, practical implementation of this type of encoding is much simpler. Instead of dealing with isolated bits reading one to the left from the particular quantization level (threshold), and zero to the right, the position of the one-to-zero transition can be used as the only information carried by this binary vector. Using thresholds instead of elementary bits as input information to the adaptive logic networks greatly simplifies the input of the data.

Table 2.7.1. Basic characteristics of different types of binary encoding.

ENCODING TECHNIQUE	Number of bits in binary vector	Error due to decoding
binary representation	$n = \log_2 w$	large
random walk	$n > \log_2 w$	small but uncontrollable
unary	$n = w - 1$	small and controllable

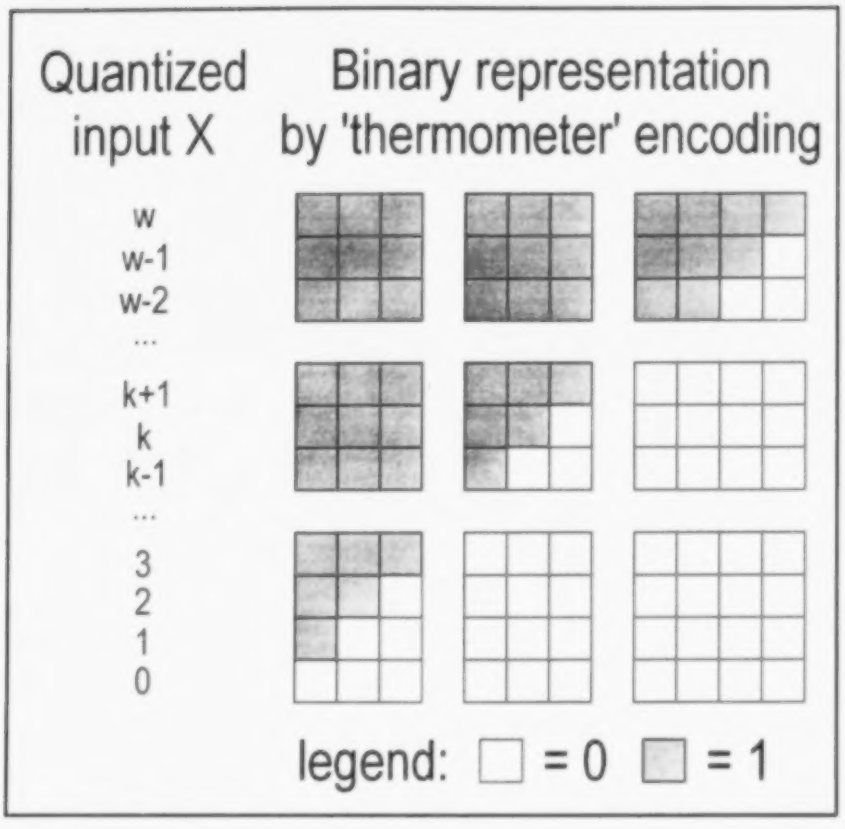

Figure 2.7.5. 'Thermometer' or unary encoding.

2.7.2 Inductive learning

Since the inductive learning algorithm has already been evaluated in problems similar to one addressed by this thesis (Kirkwood and Andrews, 1989), it was interesting to do a feasibility study using this technique and adaptive logic networks in parallel with the same data and to compare the results. Inductive learning is a symbolic machine learning technique which uses a supervised learning method as well and results in generating classification (decision) trees that are based on the input data. The algorithm is based on the hierarchical mutual information classifier of Sethi and Sarvarayudu (1982). The algorithm was implemented by Andrews et al. (1989) and Kirkwood (1989) in a program entitled DISCIPLE, which was later modified by Heller (1992). Heller's main

improvement to the existing program was the addition of a fuzzy weighting to input signals. His program Empiric, originally developed for the IBM PC, was used in this thesis to implement the inductive learning algorithm. As the goal of this study was to compare two learning algorithms, the capability of the program to assign fuzzy class membership to the output classes was not used. An example of a simple decision tree made by the inductive learning technique is presented in Figure 2.7.6.

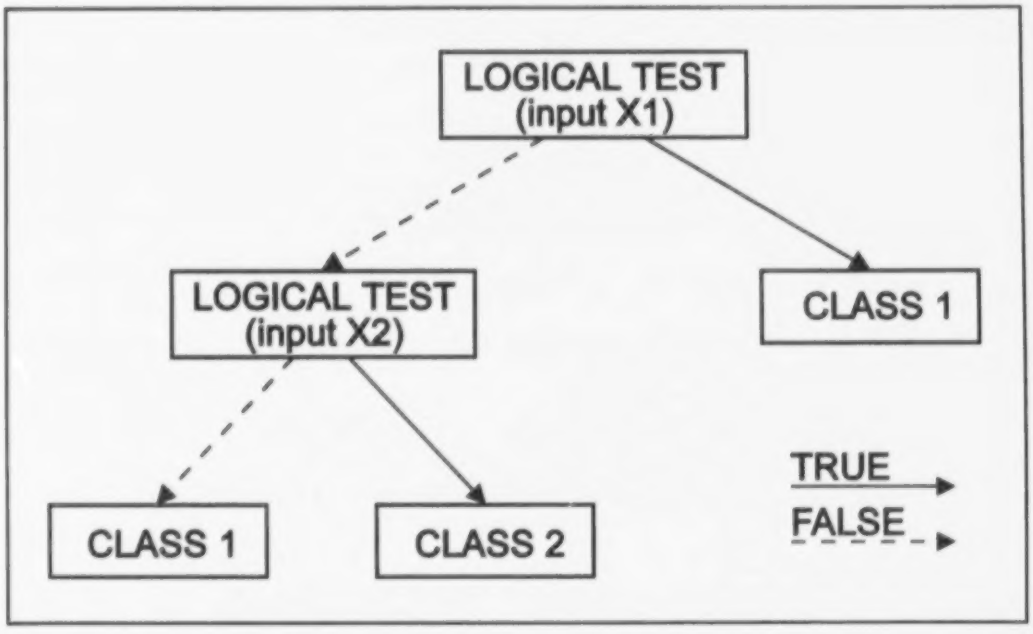

Figure 2.7.6. Inductive learning technique produces simple and comprehensible decision trees. X1 and X2 are input attributes (whose possible pairs of values represent the domain of the function to be learned). The pairs are classified into two classes (representing codomain).

The theoretical background of this machine learning technique is explained in detail in Appendix B. Briefly, this implementation of inductive learning is able to operate on a set of input signals (up to 16) representing domain, having only one output representing codomain. While generating the decision tree, the algorithm performs a hierarchical partitioning of the domain's multidimensional space. Each new node of the decision tree contains a rule based on a threshold of one of the input attributes. Each new rule further subdivides the example set. Training is finished when each

terminal node contains members of only one class. Since the number of output classes is equal to the number of quantization levels in the output signal, it is not desirable to have too many levels or the tree may become large[5]. Training may result in large trees also if the task is to learn the training set perfectly. In such a case, the learning algorithm, after extracting all invariant characteristics from the training set, continues to train on the noise. This algorithm uses real numbers, so that encoding is not used, which simplifies the program.

The characteristics that make this learning algorithm attractive are:

- relatively small decision trees,
- fast learning, and
- the rules are explicit and comprehensible.

The critical characteristic of an inductive learning technique is the size of the decision tree. As the tree grows, all the positive features listed above become invalid because the learning algorithm tries to learn noise and the speed of the learning process decreases while the decision trees become incomprehensible despite the fact that they are still explicit. The training for IL is usually very fast (within 10 seconds on an IBM PC 486DX/50MHz machine for 1000 samples).

---

[5] The ALN V.3 uses a DTREE which is similar decision tree to IL, but instead of producing a constant value at the leaves, it produces a simple expression involving linear functions and maximum and minimum operations. This expression must be evaluated to get the output value.

2.8 SOFTWARE SUPPORT

2.8.1 Evaluation of ALN V.2 - program 'WALKON'[6]

Implementation of adaptive logic networks in control requires all of the data preparation and MLT-related phases illustrated in Figure 2.7.1. However, to evaluate adaptive logic networks learning module Atree versions 2.x (denoted in further text as ALN V.2) for application in control of biomedical systems, a program entitled 'WALKON' was developed. The development of the program started as a transfer of an existing C-based program code from the UNIX-platform, to the IBM PC environment. A relatively small amount of operating RAM (Random Access Memory) and the lack of virtual memory in IBM PC computers operating under MS DOS was the main reason for deciding to base the further development on Microsoft Windows. Initial work on the program was started when newest release of MS Windows was version 3.0, and the latest version of the program was produced for MS Windows V.3.1. Two major versions of the program were produced, WALKON V.1 and WALKON V.2. A commercially available program Axotape was used for data acquisition and storage, which significantly shortened the time required to obtain the first encouraging results. The program WALKON is described in detail in Appendix C. Briefly, the concept of the program is presented here.

The program contains all functions that are not shown in gray in Figure 2.8.1. It can retrieve and save ASCII files in a specific format, which will be called for simplicity 'Walkon' format. This format is also described in Appendix C. The program can also import data from ASCII files prepared in matrix form, where the signal channels are organized in columns separated by Tab character, and samples in rows.

---

[6] The author's contribution in production of program WALKON is as follows: setting the program's concept, specifying the program's functionality, defining preprocessing functions, testing the program throughout the whole programming process, and providing active consultations to Mr. A. Dwelly (WALKON V.1.) and Mr. M. Thomas (WALKON V.1 and V.2), who did most of the programming work.

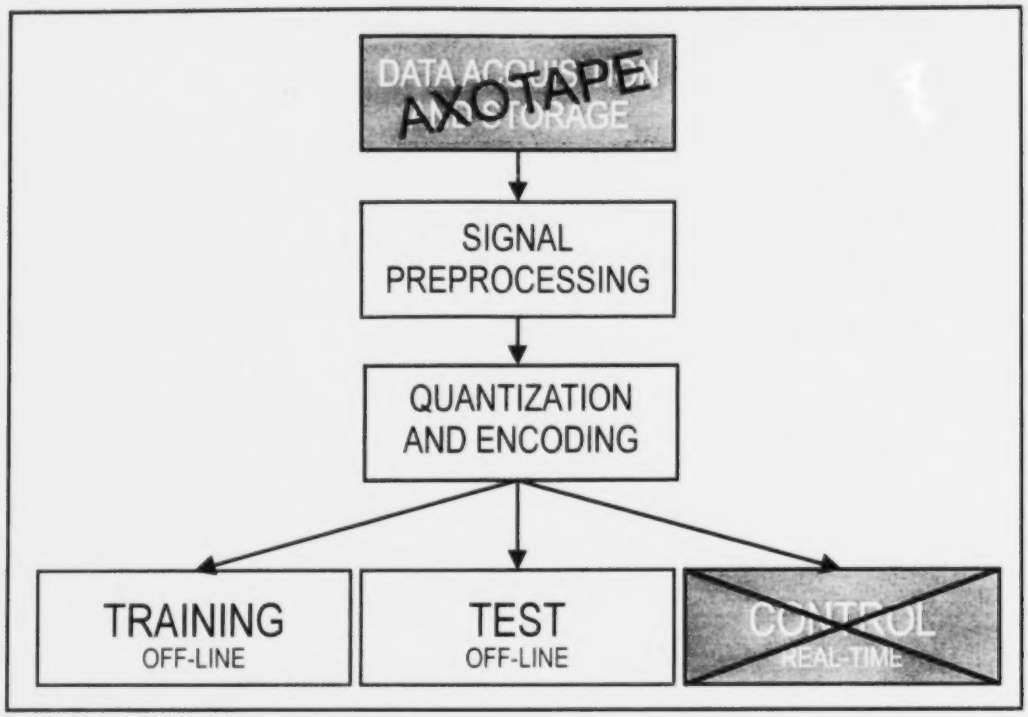

Figure 2.8.1. Data preparation and MLT functions built into program WALKON, which is developed for evaluation of ALN V.2 functions.

Program controls and commands can be accessed in two different ways: all of the functions can be accessed through standard and customized MS Windows menus and some of the most often used functions can be activated by selecting an appropriate icon on the customized toolbar. The WALKON program has an extensive context-sensitive on-line help.

After the data are retrieved, the functions and parameters of the signal preprocessing, quantization and encoding become available in the form of a customized spreadsheet. Signals are called channels and they are organized in rows of the spreadsheet. The characteristics of a single channel and the functions that can be performed on the same channel are organized in rows. Those functions that can be performed on multiple channels are grouped under menu title

'Channel'. Signals can be diagramatically displayed in separate windows or stacked together in a single window.

Standard MS Windows functions grouped under the menu *'Edit'* apply to channels. They enable the user of the program to *Cut*, *Copy* and *Paste* channels, one at a time.

To define the input data set, each channel should be characterized either as *Domain*, *Codomain* or *Not Included*. Previous samples in a single channel can be defined using the function *Set Hpoints* located in the corresponding row of the spreadsheet. Previous samples in all channels can be defined using the menu function *File History Points*. Parts of the data set can be cut out using the function *Truncate Channels*. Truncation is reflected to all channels due to the restriction that all data samples contain data from the same channels.

A type of encoding and related parameters can be specified using the function *Encode*. Two types of encoding are available: *Random Walk* and *Unary* (called *Linear* in the program). After the type of encoding is chosen, various encoding parameters can be applied to different channels or all channels can be encoded the same way using the *File Encode* function. In addition to uniform distribution of the quantization levels, *Linear* encoding allows for arbitrary positioning of quantization levels.

To produce various delays between *Domain* attributes and the *Codomain*, which is useful for prediction of future events, there is a function *Shift*. Another function, called *Differentiate* replaces the original signal in the selected channel with its time derivative. To produce a differentiated channel and to keep the original one is possible by reproducing (*Copy* and *Paste*) the original channel before one of the copies is differentiated.

After the data set is prepared for training, a *New* ALN can be selected from the *'ALN'* menu. Training parameters are available in two dialog windows. In the first one the number of presentations of the data set to the learning algorithm, called *'# Epoch'* and the allowed training

error, called *'% Correct'*, are selected. The second dialog window gives access to the *Function/Tree Parameters*. Here, the user specifies *'# Leaves'*, which defines the size of each binary ALN tree, and *'# Voters'*, which defines the number of parallel trees that will be trained to produce the same *Codomain* bit. Both windows list all *Domain* and *Codomain* channels and still allow for change of channel status (*Domain, Codomain, Not Included*). After all training parameters are decided, training can be started. During the training, a message window reports on the current training *Epoch* and the training success. As soon as the number of correctly learned bits reaches the preset training error, the training is stopped and the training time is displayed in the window.

In addition to training new ALNs, the program can be used to retrain existing ALNs with the same or a new data set.

To assess the performance of the trained ALNs on the training set, they can be evaluated using the function *Evaluate*. This function produces a new data set and a new spreadsheet which consists of only one channel automatically named *'Result: actual_channel_name'*. A signal representing an approximation to the desired output, produced by the ALNs is placed in that channel. To compare the original *Codomain* signal and the one produced by the ALN, there is a function *Compare*. This function can operate only on one data set, i.e., on the active spreadsheet. Since *Result* signal and the original *Codomain* signal do not belong to the same data set, either one should be copied from its data set (spreadsheet) to another one. The function *Compare* produces a new channel with a signal representing the difference between the original *Codomain* and the *Result*. The error can be measured using the function *Threshold* under the *Channel* menu. This function operates on a single channel and counts all samples whose amplitude is higher than a high threshold and lower than a low threshold. High and low thresholds can be set arbitrarily.

To test the performance of the trained ALNs on new data, a test set can be loaded and the ALNs evaluated in the same way.

2.8.2 ALN V.2 and ALN V.3 implementation in the real-time control: program 'FESCONT'[7]

To choose a computer platform for integration of machine learning techniques into the control system, as illustrated in Figure 2.7.1, the following criteria were followed:

- the computer platform must run Windows, because ALN learning modules for Windows already existed;
- hardware and software components should be 100 % compatible;
- programming in the new environment should be simple and flexible.

An IBM PC equipped with an AT-MIO-16DH data acquisition and I/O board was chosen for the hardware platform and LabVIEW (LV) for Windows (National Instruments, 1993) was chosen as the programming environment which will be used to integrate the whole control system. LabVIEW offers a graphical programming environment for constructing application software for the PC. It has an extremely short learning curve compared to classical programming environments.

The program FESCONT was designed which contains all the modules illustrated in Figure 2.7.1. The structure of the program follows the concept of 'virtual instruments', i.e., the computer simulates a highly customized self-standing instrument. The program is still in its development phase (functional prototype) and will not be presented in detail. The main functions of the program are listed in Table 2.8.1.

---

[7] The concept of this program was developed completely by the author. All of the program modules programmed in LabVIEW for Windows have been produced, tested and integrated by the author. The ALN learning module and interface module for communication between Windows and LabVIEW were provided by Dendronic Decisions, Ltd. of Edmonton, with particularly valuable assistance by Mr. Monroe Thomas.

Table 2.8.1. Main functions of the program FESCONT and their description.

CREATE CHANNELS	Used to define data sets which will be used in the experiment
DATA DISPLAY	Displays signals without recording (computer simulates oscilloscope)
DATA ACQUISITION	Records signals in binary format to a computer disk
ALN TRAINING & TEST	Used for ALN training and ALN test on stored signals
ALN CONTROL	Applies stored ALNs to the real-time control problem

As previously mentioned, during the course of this thesis project, Dendronic Decisions, Ltd. made a significant improvement to the ALN learning algorithm, which resulted in switching from ALN V.2 to ALN V.3 just before the control system was integrated. The program exists in two versions, one which uses ALN V.2 learning and interface modules and the other one which uses ALN V.3. The difference in the program is just in function calls which are slightly different for the two versions. Since the most recent results in this thesis are done with the ALN V.3, ALN function calls for that version are listed in Appendix C.

The program is modular, which means that the learning module can be easily replaced with another one. This provides a frame for standardized evaluation and comparison of different machine learning techniques for supervised learning.

Only two modules were not developed in the LabVIEW environment, but in the Microsoft Visual C++ environment: ALN and LV-ALN interface module. These modules were developed in the form of Windows DLLs (dynamic-link libraries) by Dendronic Decisions Ltd. LabVIEW for Windows is a 32-bit integrated programming environment and Windows 3.x is a 16-bit operating system. This discrepancy caused incompatibility between LabVIEW programs and custom Windows DLLs. To resolve this incompatibility, a third-party program Downshift (Viewpoint Software Solutions, 1993) was used to bridge between the two environments. Future versions of MS Windows (MS Windows 95) will be based on a 32-bit environment, which will be a more elegant solution of this small problem.

2.8.3 Program 'EMPIRIC'

To evaluate the inductive learning technique, program EMPIRIC was provided by its author Ben Heller. The program is designed to operate under Microsoft DOS on data prepared in specific format described in Heller (1992).

3. RESULTS

3.1 MANUAL CONTROL OF FES FOR LOCOMOTION

FES-aided walking was introduced in the rehabilitation of selected spinal cord injured subjects gradually. This process can be considered as having three distinct phases:

1) In the first phase the subject has to increase the physical strength of the whole body, which can be significantly affected by reduced mobility during hospitalization. Muscles under voluntary control are exercised through an appropriately structured work-out program, and paralyzed muscles are stimulated. The subject becomes familiar with basic FES principles and the use of the FES-system.

2) Using the appropriate mechanical aid (parallel bars, harness, frame, four-point walker), the subject practices standing. The subject is also introduced to FES. After the subject has accomplished independent safe standing for three to five minutes, the periodic movements needed for walking are introduced and exercised. The subject learns how to operate the switch(es) to start and stop stimulation.

3) The subject's gait training starts by extending the walking distance from several steps between parallel bars to as many steps as he or she is comfortable with using the mobile mechanical walking aid. In the beginning, switching of the stimulator is usually done by the therapist. However, the subject is encouraged to learn the appropriate switching for walking as soon as possible, which finally provides the subject with more independent functional walking. The switches are usually installed on the mechanical walking aid handle or on the fingers of the more functional hand. Such a position enables the subject to operate the switches with minimal movements of the rest of the body.

The goal of functional walking is not only to walk long distances and provide a greater blood supply to the paralyzed extremities, but also to be mobile and independent in narrow and tight spaces, such as in a kitchen or bathroom.

Before the first gait trial, the subject should be reevaluated to reduce (or eliminate) the risk of sudden pathological changes. The most important criteria appear to be limited range of motion, osteoporosis or skeletal system instability. Any of these factors may exclude the subject from gait training.

Subject L.W., who is presented in detail in the 'Subjects' section of Methods and Materials, started FES-assisted walking before the begining of this thesis project. She was selected as a good candidate who could benefit from automation of stimulation control. In addition, she was very interested in the opportunities that automatic control offered to her, which motivated her to dedicate a significant amount of time and effort to this project. At the beginning of this project, the subject used a single channel Mikrofes stimulator with surface electrodes placed on the skin above the peroneal nerve in popliteal fossa region. Stimulation has been used to activate the flexor withdrawal reflex on the left leg through the afferent part of the common peroneal nerve. The other (right) leg has limited flexion, and stimulation was not needed, so that leg will be denoted below as 'normal'. After the walking program started, another specially designed single channel device was received, which was able to supply enough energy for a fully implanted stimulator already placed on the common peroneal nerve in the same region.

Technically, the simplest control system for FES-assisted locomotion is manual switching of the stimulator by the subject. Since the subject is not able to walk at all without stimulation, the manually controlled gait could not be compared to no-stimulation gait. Techniques developed later as a part of this thesis are compared to manually controlled gait. The most complex FES system should be able to control stimulation timing and intensity for multichannel stimulation using multiple sensory inputs (natural and artificial).

Using manual control of FES, the subject makes the decision to start stimulation. A manual switch is used by the subject to begin stimulation and is placed on the right handle of the wheeled walker.

Taking a step, which is an automatic proces for people having normal voluntary control over their extremities, is a very complex proces for someone whose extremities are paralyzed. In this particular example, to take a step, our subject had to perform at least ten different actions to assure that her posture and the position of the walker provided safe movement at every instant of time during walking. The walking session usually starts with extensive stretching of both legs and arms, for which the subject gets help from another person. The stretching usually reduces the spasm in her extremities, which produces unwelcome clonus during walking. After the preparation is completed, electrodes (or transmission antenna) are installed and the stimulation is tested, the subject is ready for walking. She stands up from the wheelchair without stimulation, turns ON the stimulator and starts walking with her normal (right) leg first. The actions needed to walk one full gait cycle can be distinguished as follows:

- make sure that the disabled (left) leg is fully extended and can support the whole body weight together with hands supported on the walker;
- shift body weight onto the disabled (left) leg;
- advance the walker;
- take a step with the normal (right) leg;
- fully extend the normal (right) leg and make sure that it can support the whole body weight together with hands supported on the walker;
- shift body weight onto the normal (right) leg;
- advance the walker;
- press and hold the switch to stimulate the disabled (left) leg;
- take step with stimulated (left) leg, visually follow its trajectory to prevent it from hitting the other leg or the leg of the walker and from landing in an inappropriate position[8];

---

[8]This usually happens if the stimulation duration is too long or the stimulation intensity is too strong.

- at the end of swing phase, release the switch to stop stimulation and let the left leg land on the ground;
- fully extend the disabled (left) leg and assure the left leg can bear weight,
- shift the weight to the disabled (left) leg, either completely (to continue walking) or partially (to stop walking and continue standing).

The two most important and hazardous phases of this procedure are weight shifting to and from the disabled leg, and they usually take most of the subject's attention during walking, independent of the way the walking is controlled. These are the moments when it is not very clear which leg is in charge of being the main body weight supporter. It is obvious that the number of unusual tasks to perform during each gait cycle is too large to expect a fast and easy training procedure. Not a single operation from the above list was simple to learn. Every action required the full concentration of the subject and much effort and energy to perform. The walking was very slow. It provided the subject with all the positive effects of the use of FES, but it was quite far from fully functional walking.

The maximal distance per trial was 15-50 m, which was equivalent to 30-100 steps. There was a wide range of maximal distances per trial because many factors affected mental and physical fatigue. The subject's very fast fatigue is the main reason for very short maximal walking distance per trial, which also depended very much on the type of floor surface, time spent stretching and exercizing before walking sessions, etc. Although the manual switch is located at a very convenient place on the walker so that operating it didn't require any movement of the hand, the switching task is an additional burden on the right hand, which was also the major support of the body weight on the walker. This was very tiring, particularly in the learning phase, before pressing on a switch became a part of a routine walk.

The stimulation intensity required to take a step with the disabled (left) leg was not constant for all walking sessions and during any particular walking session. Very often, if the intensity was close to eliciting the flexor withdrawal reflex response, but still too low, prolongation of the stimulation

duration was a way to activate the reflex. If longer stimulation didn't help, adjustment of the stimulation amplitude was needed. However, the limiting factor with the first stimulator the subject used (with surface electrodes) was the pain the subject experienced if stimulation intensity was too high. A fully implanted stimulator eliminated this discomfort because pain receptors in the skin were not being stimulated. In addition, its stimulation was less amplitude-sensitive and stimulating conditions were more constant.

Surface stimulation intensity also depended on the changes in electrode-skin contact impedance, as well as on the active electrode position. The advantage of this system was that, once electrodes are put in the right position, they can stay there for two to three weeks. Fully implanted stimulation intensity depends on the position of the transmitter's antenna because of very limited operating range (1 to 2 cm). However, the adjustment of the antenna's position is easy because it is held in place by an elastic strap.

Gait cycle stimulation duration was estimated by the subject visually following the trajectory of the stimulated leg during the swing phase. If stimulation intensity was not high enough, longer stimulation was needed and sometimes this longer stimulation would elicit a reflex response. Otherwise, the stimulation amplitude was increased one 'notch'. This was required quite often during long distance walking trials with surface stimulation.

Manual control by a skilled subject or therapist is the most reliable type of control of FES-assisted locomotion available. There is no need to translate, or transduce signals from the subject to the stimulator to detect when and for how long she wants to be stimulated. The subject does it all and explicitly informs other parts of the system about her intentions. There is no missing stimuli when the subject wants to be stimulated, and extra stimuli are very rare - they occurred only in long walking sessions when the subject's concentration decreased and the muscles driving the thumb, which switches the stimulation ON and OFF, became fatigued. Misfirings were extremely rare with manual control of the stimulation.

Despite having important advantages, such as reliability, the manual system showed a few disadvantages. It didn't provide the subject with a more functional walk, with the possibility to improve quality of walking or to reach far points with less energy, i.e., to walk more efficiently. It was obvious that the subject had reached the achievable maximum of performance in manually controlled walking. Further improvement in locomotion required stabilization and reduction in the duration of the stance phase, which could be achieved by replacing the subject in carrying out some of the complex actions necessary for FES-assisted walking by implementing them into an 'intelligent' automatic controller. The goal in designing an automatic control system for FES-assisted walking is to preserve or even improve the reliability and safety of a manual system and to bring more functionality and more efficacy to the disabled gait. One way of achieving this goal is by duplicating the criteria used by the subject in switching the stimulation ON and OFF.

3.1.1 GAIT ANALYSIS FOR MANUAL CONTROL OF STIMULATION

Performing gait analysis is one of the best ways to understand the subject's walking, and to follow the progress in rehabilitation or in learning of control functions.

To evaluate different stimulation configurations (different number of stimulation channels, different ways of applying the stimulation), a gait analysis method proposed by Winter (1991) was implemented in computer programs SELPOS and ENSAVG written by Marc Bélanger (Bélanger, 1992). Joint angles in the sagital plane from hip, knee and ankle were recorded using Penny and Giles electro-goniometers during FES-assisted walking and averaged over the gait cycle. After being marked, each gait cycle was divided in 100 bins, normalized and averaged. The gait cycle is considered from 'toe on' at 0% to 'toe on' at 100% (the subject touched down first with her toes on both legs). Figure 3.1.1 compares joint angles between single-channel and three channels of surface stimulation and one channel implanted stimulation. These kinematic measures showed that walking was slow (A), but could be improved by adding extra channels of stimulation to the right gluteal muscles to stabilize the right knee and hip joints, while the left leg went through the

swing phase (B). The extra channels of stimulation increased the speed of walking, without changing the stride length (Stein et al., 1993). The very long stance phase was shortened from 5.7 to 3.9 s and the swing phase from 1.9 to 1.4 s. The steps also became more regular (compare the standard error bars for the hip and knee early in the stance phase of Figures A and B.

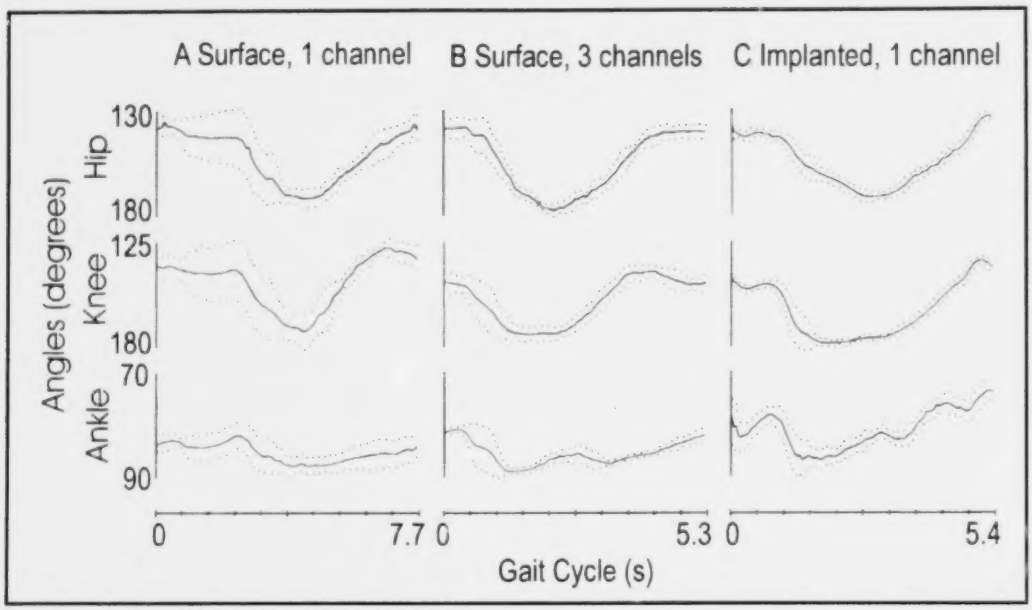

Figure 3.1.1. Gait analysis of a subject L.W. walking with (A) a single channel of stimulation applied to the skin over the common peroneal nerve, (B) two additional channels applied over the quadriceps and gluteal muscles and (C) a single channel stimulator surgically implanted over the common peroneal nerve. Averages (solid lines) and standard errors (dotted lines) have been computed for the hip, knee, and ankle joint angles for 5,5, and 9 gait cycles in (A), (B), and (C) respectively. Each gait cycle began at the time of foot contact. For all angles, flexion is displayed upward with respect to the straight (180°) or right angle position (90°). (modified from Stein et. al, 1993)

It was already mentioned that the subject touched the ground at the end of her swing with her forefoot. This fact made it interesting to record ground reaction force under the front of her healthier foot, which could be used for control of the other, more disabled leg. This measurement was achieved by a force sensing resistor (FSR) installed under the ball of her right foot. Averaged signals recorded from one FSR and the manual switch used to control timing of the stimulation during walking are presented in Figure 3.1.2. From these recordings it became obvious that the

stimulation usually starts some time after the right foot finishes its swing and touches the ground. This was also expected, because the subject never started stimulation before she was sure that the right foot was on the ground and in full extension, providing safe body weight support during the swing of the stimulated leg.

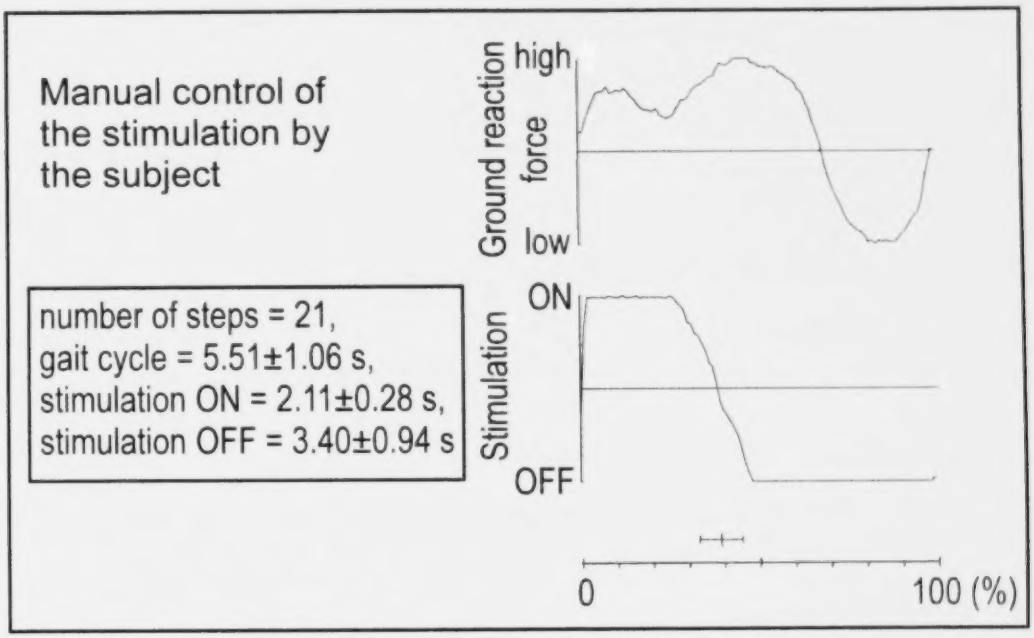

Figure 3.1.2. Ground reaction force measured under the ball of subject's right foot presented together with the switching signal recorded from the manual switch used to control FES during walking. The FSR used for measuring ground reaction force is highly nonlinear, so only a qualitative scale is shown.

In addition to these kinematic and kinetic measurements, timing of the events in the gait cycle was studied using foot-switches under the heel and ball of the feet of both legs. Figure 3.1.3 shows two combined signals recorded from foot-switches installed under heel and ball of the foot of each leg with corresponding signal representing manual stimulation control. Switching of the heel switches is represented by smaller increments and switching of the switches in the front of the foot are represented by larger increments. Signals representing two foot-switches from each leg are combined into one signal. These signals confirmed findings from video records and ground force

measurements about the sequence of events during the gait cycle: the subject touched the ground with her toes first in both legs.

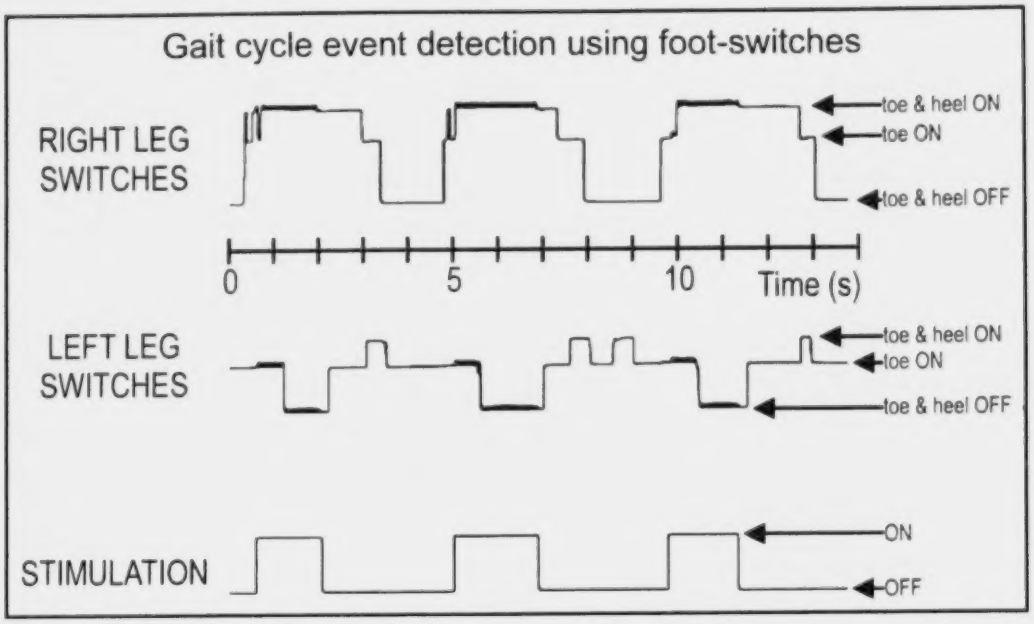

Figure 3.1.3. Detection of the events during FES-assisted gait. The first two traces are combined signals recorded from switches installed under the heel and under the ball of the foot of right and left leg respectively. The third signal represents manual stimulation control.

3.2 HAND-CRAFTED RULE-BASED CONTROL OF FES[9]

After the subject had reached the rehabilitation phase in which she was able to ambulate with manual control of the stimulation, the next phase of the development was to automate the FES-control and decrease the cognitive burden resulting from unnatural starting of the swing phase of her more disabled leg. Automation of the control should not introduce more problems than benefits for the subject. The implementation of this strategy resulted in the use of a rule-based system to mimic the manual switching control of the stimulation operated by the subject herself or by a skilled physiotherapist. Extensive gait analysis was done to uncover the rules which may best explain the way a skilled person decides to start and stop the stimulation bursts before and during the swing phase of the disabled leg. To detect the current state of motion of the subject, potential feedback signals were recorded from:

- foot-switches, explained in the previous section (Figure 3.2.1);
- insole force sensors, installed in the subject's shoes measured changes in the force distribution under the front (metatarsal area) and back (heel) of both feet during ambulation (four sensors) (Figure 3.2.2); and
- flexible goniometers attached across hip (flexion-extension and abduction-adduction), knee (flexion-extension) and ankle (flexion-extension and inversion-eversion) of a disabled leg measured changes in joint angles (three transducers with five sensory outputs).

In addition, a binary signal was recorded indicating the time intervals when the person operating the control pressed on the manual switch. The subject was also video-taped during walking.

---

[9] A version of this chapter has been presented at the IFAC Symposium on Modeling and Control in Biomedical Systems, in form of paper "Improved Methods for Control of FES" by Kostov et al., 1994. The paper won a Student Research Award at the meeting and it was published in the conference proceedings.

After the examination of the recorded data, we noticed that the initial trigger starting the stimulation of the disabled leg for its swing phase may come from a single sensor on the other leg, i.e. taking a step with a 'normal' leg may be interpreted as the signal of intention to take a step with the disabled leg too. Basic statistical values related to this strategy have been calculated over several walking trials. This analysis resulted in the following values of parameters (mean ± SD) illustrated in Figure 3.2.1:

- ***T1*** - a delay from the time when the 'normal' leg touches the ground to the start of the stimulation (0.22 ± 0.09 s);
- ***T2*** - a time interval during which the subject holds the switch down, i.e. the duration of the stimulation (1.51 ± 0.18 s);
- ***T3*** - a part of the stimulation duration during which the disabled (stimulated) leg is in swing phase (0.75 s ± 0.17);
- ***Tc*** - a gait cycle duration (5.51 s ± 1.06)
- ***Tv*** - a recruitment phase during which the stimulation has begun but the muscles are still not enough recruited to lift the foot off the floor and produce the swing of the leg.

Even very basic statistics (see Table 3.2.1) uncovered regularities which could be used to define timing for 'automatic' FES-control, simulating manual switching control. Since the subject didn't have voluntary control over hip flexion of the disabled leg, a signal to start new step stimulation couldn't come from that leg. The normal leg was found to be able to signal a new gate cycle with the time of its contact with the ground. The intention to take a step with the disabled leg could be detected by detecting the fast positive transition in force measured from the front of the normal foot. The stimulation could be started T1 s after the normal foot touches the ground and finished either after a preset duration T2 or after the delay T3 after the ipsilateral foot leaves the ground (stimulated leg swing phase - ipsilateral 'toe OFF'). Since the subject touches the ground with the forefoot rather than with the heel, one switch under the ipsilateral toe was enough to define the stimulation ON time.

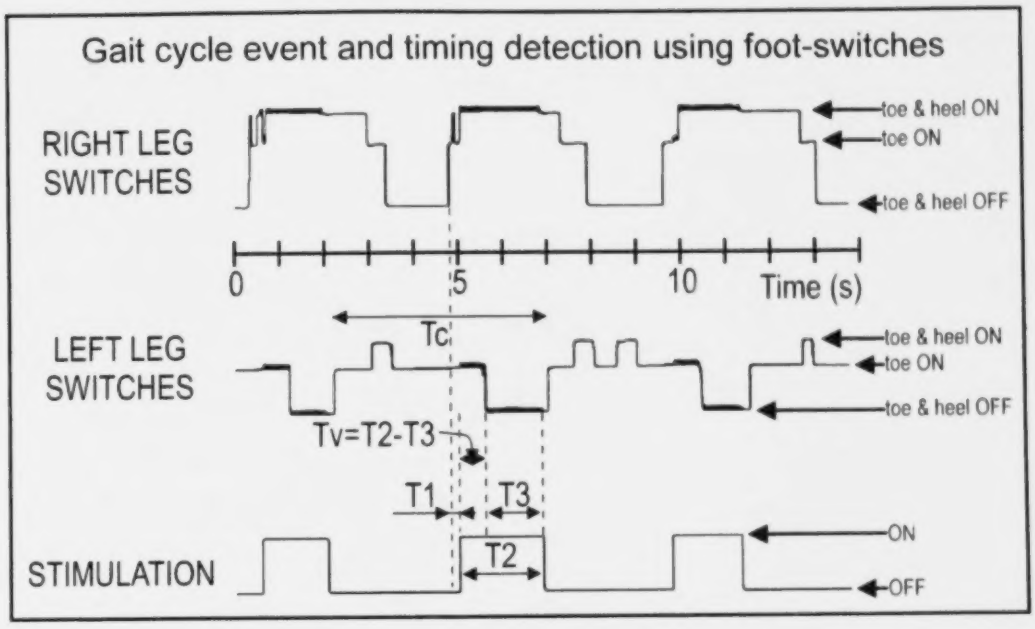

Figure 3.2.1. Gait cycle event and timing detection using foot-switches. The first two traces are combined signals recorded from switches installed under the heel and under the ball of the foot of the right and left leg respectively. The third signal represents manual stimulation duration control. Signal levels for 'toe ON', 'toe & heel ON' and 'toe & heel OFF' are marked as well as time intervals which are used to define relationships between the start and duration of the stimulation and the position of the feet.

Table 3.2.1. Statistical values of the timing of major events in a gait cycle related to the switching control of walking (number of steps N=17).

	T1	*T2*	*T3*	*T2-T3*
mean (s)	0.22	1.51	0.75	0.76
st.dev. (s)	0.09	0.18	0.17	0.07
st.error	0.02	0.04	0.04	0.02
max (s)	0.42	1.90	1.10	0.92
min (s)	0.10	1.24	0.52	0.66

Values obtained by statistical analysis of the gait parameters and two different possibilities to stop the stimulation provided the basis for the development of two hand-crafted control modes called Control Mode 1 (CM1) and Control Mode 2 (CM2). CM1 has a fixed duration of the stimulation,

while in CM2 the duration of the stimulation is divided into two parts: *a variable duration phase (Tv)* during which the stimulation begins and produces the start of the swing phase; and *a fixed duration phase (T3)* which begins with the stimulated leg leaving the ground and which should be long enough to enable the swing of the stimulated leg for a range of walking speeds. To detect the lift of the stimulated leg, a switch was needed under the stimulated leg. Because of the 'flat-foot' pattern of walking, rather then heel-contact, the switch was located under the ball of the foot instead of under the heel.

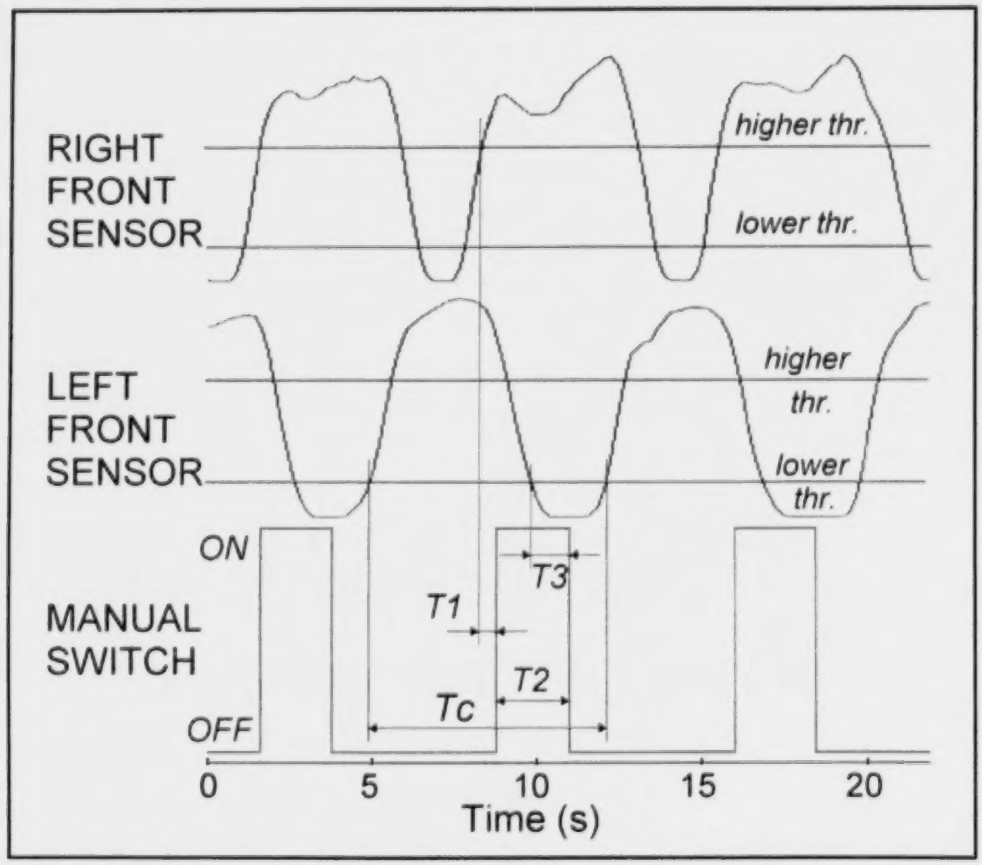

Figure 3.2.2. Gait cycle event and timing detection using only two force sensing resistors installed under the front of the foot in both legs. The first two traces are samples of the signals recorded from force sensors positioned under the central metatarsal areas of both feet. Force increases upwards. The signal in the third trace indicates the time intervals when the subject pressed on the manual switch to deliver stimulation to her left leg.

The patterns detected by the foot-switches in manually controlled walking were the starting point for experimentation with other types of transducing devices. Much more reliable results were obtained using force sensing resistors (see Figure 3.2.2). The signal they produce is continuous, so the events are determined using fixed thresholds.

The walking pattern of the subject was the determining factor in deciding to use only force sensing transducers positioned under the central metatarsal joints of both feet. Although the goniometers give better information of the dynamics of the disabled leg, they were not considered for permanent use outside the laboratory because of their high sensitivity to mechanical stress, very high price and because the process of putting them on and taking them off requires manual dexterity not present in the subject.

3.2.1 Control Mode 1

This control mode uses only one force sensor installed under the central metatarsal area of the right foot, which is considered 'normal'. The state diagram of this mode includes all states presented in Figure 3.2.3, except those that are shaded and its states are defined as follows:

- STATE 1 is entered when force on the 'normal' (right) foot exceeds a higher threshold (stable body weight bearing on the fully extended right leg). If the force drops under the lower threshold during the ***T1*** interval, it is considered that clonus in the 'normal' leg has started and the whole process is reset by returning to STATE 0.
- STATE 2 (STIMULATION ONSET) begins when force on the 'normal' foot is maintained above a lower threshold for ***T1*** seconds.
- STATE 3 (STIMULATION CONTINUES) is passed through after the stimulation duration exceeds a fixed interval ***T2***.
- STATE 4 (STIMULATION OFF).
- Return to STATE 0 when ***T4*** exceeded (no clonus) and force is less than the lower threshold (swing phase of 'normal' leg).

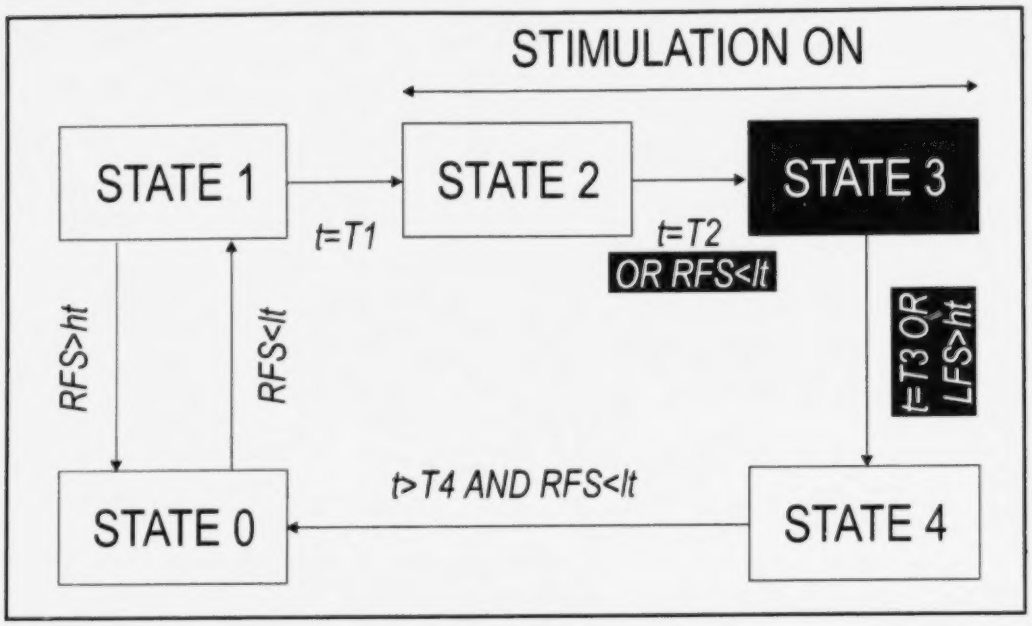

Figure 3.2.3. State diagram and transitional conditions for Control Mode 1 (CM1) and Control Mode 2 (CM2). CM1 includes all states and transitions except these in the shaded boxes. CM2 includes all the states and conditions presented. RFS ***and*** LFS ***stand for Right and Left Front Sensor and*** ht ***and*** lt ***stand for the higher and lower thresholds respectively.***

3.2.2 Control Mode 2

The second control mode employs two force sensors: one sensor installed under the central metatarsal area of the right foot and another one installed under the central metatarsal area of the disabled leg. The state diagram for this mode includes all states presented in Figure 3.2.3 and its states are defined as follows:

- STATES 0, 1, and 2 are defined in the same way as in CM1.
- STATE 3 (STIMULATION CONTINUES) begins when force under the disabled (left) foot drops under the lower threshold (swing phase begins) or the stimulation duration exceeds ***T2***.
- STATE 4 (STIMULATION OFF) occurs when the time in STATE 3 exceeds ***T3*** or the force under the disabled foot exceeds the upper threshold (standing or stumble).

The only differences between these two control modes are in STATES 3 and 4. STATE 3 of CM2 monitors the effect of the stimulation which starts during STATE 2. If it is successful, it is expected that after the stimulated foot touches the ground the stimulation is not required and should be stopped. In the case when the stimulation parameters are not optimal and the stimulation is not very efficient or if the shifting of the weight from the disabled to 'normal' leg is not finished, this feature prolongs the stimulation giving the subject more time to maneuver and to finish the step.

3.2.3 Practical implementation of Control Modes 1 and 2

Control Modes 1 and 2 were implemented in both hardware using discrete electronics and software using a microprocessor controller (Motorola MC68HC11). The block-diagram of the discrete electronics hardware implementation is presented in Figure 3.2.4. The electronic device was designed and assembled to operate as a replacement for the manual switch, i.e., it used the same control input to the stimulator. The reason for such an approach was to preserve the basic or reference control mode of the stimulation, and not to leave the subject without the stimulator during development of the electronics.

Device shown in Figure 3.2.4. has three control inputs:

1) 'RIGHT FSR' for contralateral force sensor (used in both control modes);

2) 'LEFT FSR' for ipsilateral force sensor (used only in CM2); and

3) 'MANUAL SWITCH' (used as an override control function to start stimulation in case of missing stimulus.

In addition to control inputs, the device also has precise control of four time intervals:

1) T1 - delay of the start of stimulation which is initiated by the 'RIGHT FSR' signal exceeding its higher threshold;

2) T2 - duration of the stimulation (CM1) or maximal duration of the Tv part of the stimulation as a preparation for swing phase (CM2). If the transition from high to low on the 'LEFT FSR' is not detected during this interval (most often due to low intensity of the stimulation) stimulation is cancelled.

3) T3 - duration of the stimulation during the swing phase of the stimulated leg (CM2).

4) T4 - clonus avoidance time interval after the stimulation is finished and swing phase is completed. During this interval stimulation is denied even if other conditions to start another step are fulfilled.

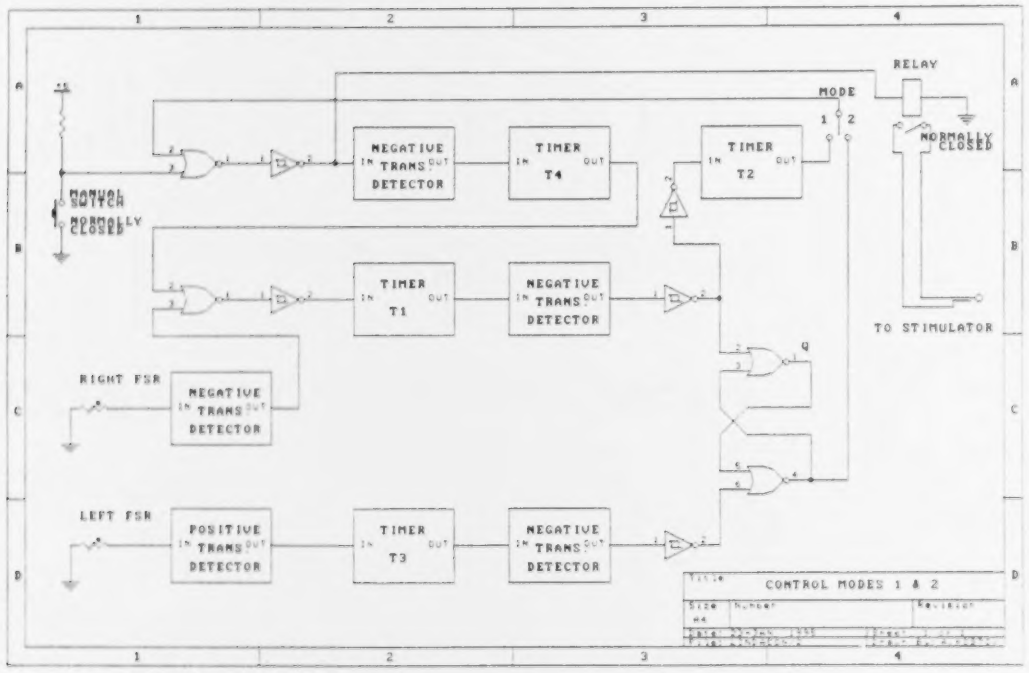

Figure 3.2.4. The block-diagram of the implementation of CM1 and CM2 in discrete electronics.

At any time, the subject is able to activate stimulation of the disabled leg by pressing on the 'MANUAL SWITCH'.

Subject has clonus on both sides. Thus, the clonus avoidance protection had to be provided in all phases of the gait cycle. Avoidance of clonus was included in both control modes. Since clonus usually occurs after the movement of the leg, this protects the subject after the swing phase in both legs. The following safety features are included in the design:

- The first safety feature is activated after the swing phase of 'normal' leg. It is active in STATE 1 and protects the subject from being stimulated when the 'normal' leg, which should take the whole body weight during stimulation of the disabled leg, develops clonus and becomes unstable. If the subject had clonus on the contralateral leg before the start of stimulation, and if it did not stay stable on the ground at least for T1 s, the stimulation was denied and the system returns to STATE 0. In that case it was presumed that the normal leg is not stable enough to support body weight. (CM1 & CM2)
- The second safety feature takes care of clonus on the contralateral leg which may start during the first phase of stimulation, and prevent the stimulated leg from leaving the ground and starting the swing phase during interval T2 (transition high to low on LEFT FSR is not detected). In such a case the swing phase stimulation for an additional T3 duration will be cancelled. (CM2)
- After the stimulation is finished, instability caused by weight shifting from the stimulated to the non-stimulated leg or by clonus on the stimulated leg may satisfy conditions to start stimulation for a new gait cycle before the subject is ready for it. In this case the subject is protected by denying the stimulation for the interval T4 s, selected to provide the subject with enough time to stabilize and to continue walking by taking next step. (CM1 & CM2)

The frequent adjustments of thresholds were an inconveniece during walking. Thus, in the next design the control modes were implemented in software, rather than in hardware. Therefore, the control logic was implemented as *a state machine* running on a Motorola MC68HC11 microprocessor controller developed by Mr. Michel Gauthier (hardware and software) and Mr. Robert Rolf (hardware) in the Division of Neuroscience, University of Alberta. This approach provided new functionality in experimenting with different control strategies.

- A micro-controller connected to a remote PC enabled us to record signals from the sensors and to experiment by varying control parameters during simulation of walking;
- A calibration routine was implemented enabling the subject to set up the threshold levels during walking. It was implemented to exclude frequent screwdriver-type adjustments of the amplitude and offset of the sensor signals as a result of using imperfect sensors;
- Digital control of stimulus intensity (pulse width) and duration was implemented which enabled the subject to regulate these two parameters more precisely and to see the actual value (either first or second) on the digital display.

The new design was smaller, more flexible and functional than the previous one. However, some of the earlier problems remained the same and some new ones were added. The sensor signal was still dependent on the physical conditions. The calibration routine was based on measurement of the 'standing force' during a very short time interval and the adjustment of the threshold levels relative to these measurements. Since the subject did not have very good control over her posture, 'standing force' measurements were quite irreproducible, and calibrations vere mostly unsuccesful. Despite existance of the calibration routine, screwdriver adjustments of the amplitude and the offset of the sensor signal were still needed. Improper operation of the controller was experienced as missing stimuli and misfiring of the stimulator.

3.2.4 Gait analysis for automatic control of the stimulation

Since the first experiments with CM1 and CM2, several subjects have successfully started using the same or a modified type of controller based on CM1. Due to the success of CM1, and the inconvenience that CM2 brings by adding another cable coming from the other leg, there are not enough measurements to prove its superiority (or inferiority) compared to CM1.

By using CM1 the subject was able to start walking, to stop, and to maintain one speed very well for swinging the disabled leg through. Although the stimulation has fixed parameters, the subject is able to walk at different speeds in a narrow range just by varying the swing duration of the 'normal' leg.

After just a few sessions of using automatic control of the stimulation in CM1, the subject L.W. succeeded in shortening the stimulation phase of the gait cycle (see Figure 3.2.5), which resulted in reduction of the fatigue and increased distance that the subject could cover without stopping. Statistical values of the gait cycle measured during manual and automatic control are presented in Table 3.2.2.

Table 3.2.2. Average gait cycle parameters for manual and automatic (Control Mode 1) control of the stimulation (Subject L.W.)

	Step cycle duration (s)		Stimulated phase duration (s)		Non-stimulated phase duration (s)	
	mean	SD	mean	SD	mean	SD
Man. Stim.	5.51	1.06	2.11	0.28	3.40	0.94
Auto. Stim.	5.14	0.57	1.36	0.04	3.84	0.57

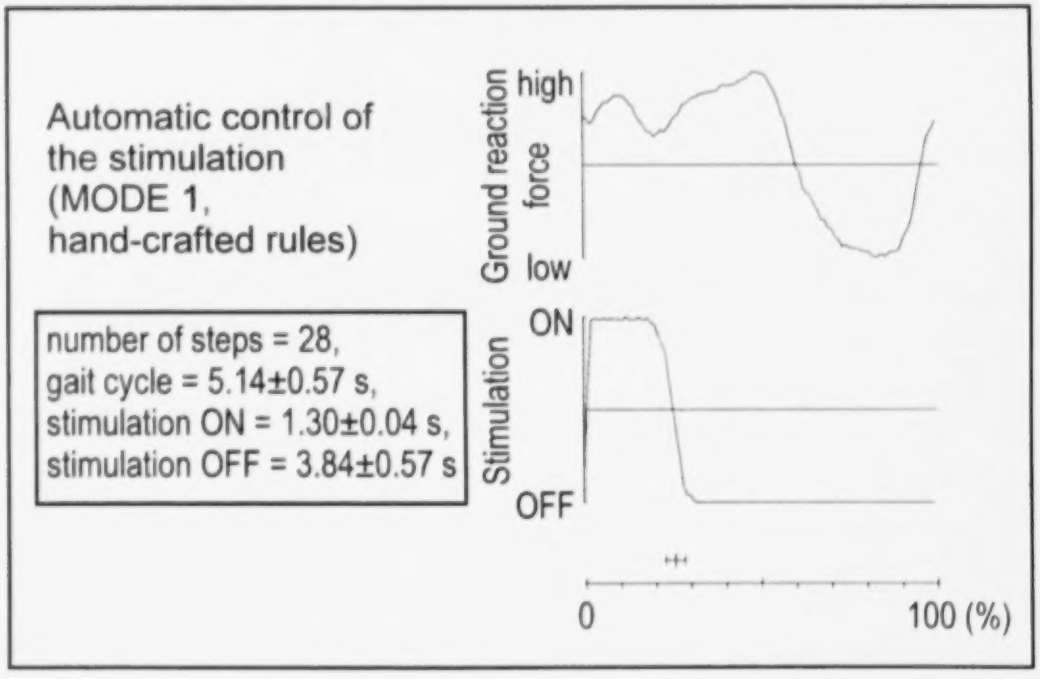

Figure 3.2.5. Ground reaction force measured under the ball of subject's right foot presented together with the switching signal generated by the Control Mode 1 logic used to control FES during walking (Subject L.W.).

3.2.5 Conclusion

Two directions were seen as most promising for further development:

1) The use of natural or artificial sensors which can reliably and reproducibly measure the kinematic and kinetic variables of walking. If the previous, present and the intended state of the locomotion are better described by signals recorded from the sensory system, it would be easier to define the actions required to acomplish the intended movement. This is specially important for protection against misfiring, i.e. having a stimulus delivered when it is not needed, because it is much easier to specify conditions for when to stimulate than those for when not to stimulate.

2) The use of artificial and/or computational intelligence systems (such as, expert systems, artificial neural networks, inductive learning algorithms, etc.) as tools to:

- overcome the sensors' imperfections;
- increase the information/noise ratio in recorded signals;
- extract control rules for stimulation control of the disabled muscles from the purified signal;
- apply a 'pattern recognition' technique on the signals recorded from the activators and nerves to monitor achieved locomotion and
- modify parameters of the stimulation or control logic (e.g. 'adaptive thresholds') if required in order to obtain a more 'normal' walking.

3.3 AUTOMATIC GENERATION OF CONTROL RULES BY MACHINE LEARNING TECHNIQUES: EVALUATION OF ADAPTIVE LOGIC NETWORKS (ALNs) FOR SWITCHING CONTROL OF FES[10]

If the system to be controlled is too complex and does not have a good mathematical model, as in control of locomotion of subjects with a spinal cord or brain injury, artificial and computational intelligence techniques may be used to generate the rules in the rule-based control system or to generate parameters for a control program that go beyond the format of rule-based control. Such a system, if designed with a significant amount of redundancy, can even overcome problems considered unresolvable, such as variability of the sensory signals, changes in walking habits, etc. One of the main advantages of this approach is its speed in generating the rules from examples.

A major task in automating the control of walking for stroke or incomplete SCI subjects is to recognize the subject's intention to take a step with a disabled leg and to provide the required control signals to the stimulator. The simplest method of control is through hand switches, but this is not appropriate for incomplete quadriplegics and stroke subjects who may not have adequate hand function. In addition, operating a hand switch requires repetitive voluntary intervention and can introduce delays and variability. Liberson et al. (1961) used a heel switch which activated a single channel of stimulation to assist in the swing phase whenever the heel came off the ground. This system does not work reliably in subjects where contractures or spasticity prevent a good heel contact with sufficient weight bearing or in subjects who suffer from clonus, which can cause

---

[10] A version of this section has been presented at the 14th Annual International IEEE - EMBS Conference in form of short paper entitled "Evaluation of Adaptive Logic Networks for Control of Walking in Paralyzed Patients" by Kostov et al., 1992. The paper won a finalist prize for IEEE Region 7 (Canada) in the Student Paper Competition.

the heel to lift and touch the ground several times during the stance phase. A rule-based system based on threshold logic applied to the signal from a force sensor installed under the toe of the normal leg has been proposed as an alternative method to detect the subject's intention to take a step (Stein et al., 1992). The duration of the stimulation was either preset or it was decided through use of another force sensor installed under the toe of the stimulated leg.

The current study is limited to investigating the prediction of FES switching commands operated by a skilled subject or physiotherapist from switching patterns in force signals measured under the feet using a machine learning technique, known as Adaptive Logic Networks (ALNs), which is a type of artificial neural network for supervised learning. If the ALN can learn and later predict stimulation switching patterns, based on biomechanical signals and manual control by the subject or a physiotherapist, then the ALN can be used to transform input sensory signals into output control signals for stimulation. The concept of such a control system, which can automatically generate the control rules is presented in Figure 3.3.1.

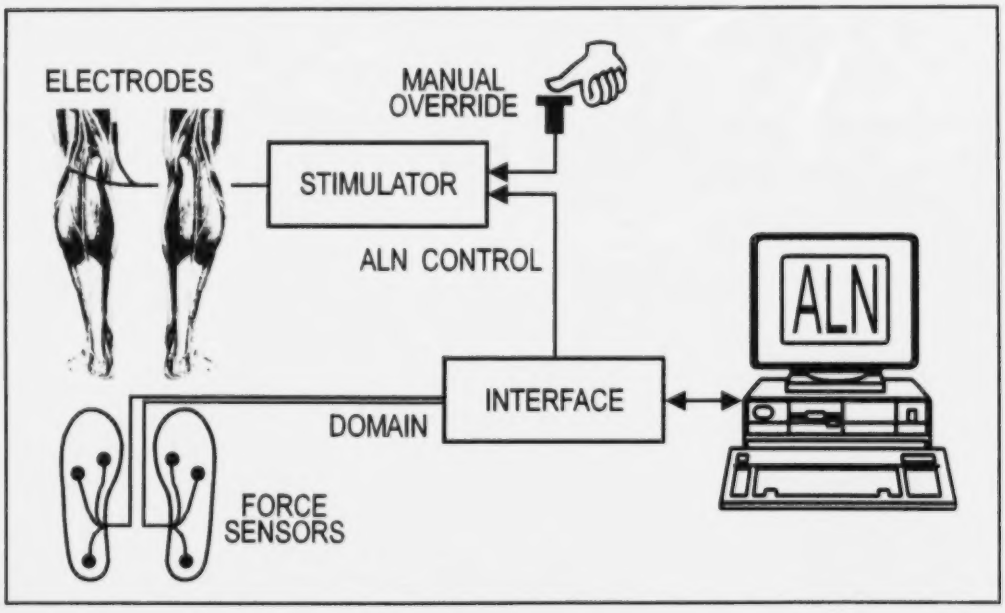

Figure 3.3.1. The concept of using machine learning techniques in control of FES-assisted locomotion.

This method is intended for subjects who are already trained to step periodically by manually pressing on the switch to turn the required stimulation on or off. Automatic control can then be added to manual control to enable the subject to concentrate on other functions during walking, such as shifting the body weight from one leg to another, avoiding obstacles, moving assistive devices, carrying objects, etc.

The subject tested (L.W.) was the same one introduced in detail in section 2.1.2.

3.3.1 Adaptive Logic Networks

Version of the ALN learning program: **ALN V.2** (see section 2.7.1 and Appendix B for details).

ALN interface program: **WALKON** (see section 2.8.1 and Appendix C for details).

Encoding technique: **Random Walk Encoding** (see section 2.7.1.1 for details).

Briefly, the result of the ALN training on a set of signals recorded from natural or artificial sensors configured as an input vector - an element of the *domain*, and corresponding outputs - in a set representing the *codomain*, is an approximation of the input-output transfer function. This function is a boolean expression that can be presented in the form of one or more trees composed of logic gates performing boolean operations (e.g. ANDs) which approximate the functional dependence between the inputs and output data (see Figure 3.3.2).

Continuous input and output values are treated by encoding into boolean vectors. To import signals from measuring devices used in the gait laboratory, an IBM PC-based program WALKON was developed which manipulates signals, converts their samples into booleans, learns input-output transfer functions and stores the learned functions in binary trees for use on future input data. The program WALKON is presented in detail in Appendix C.

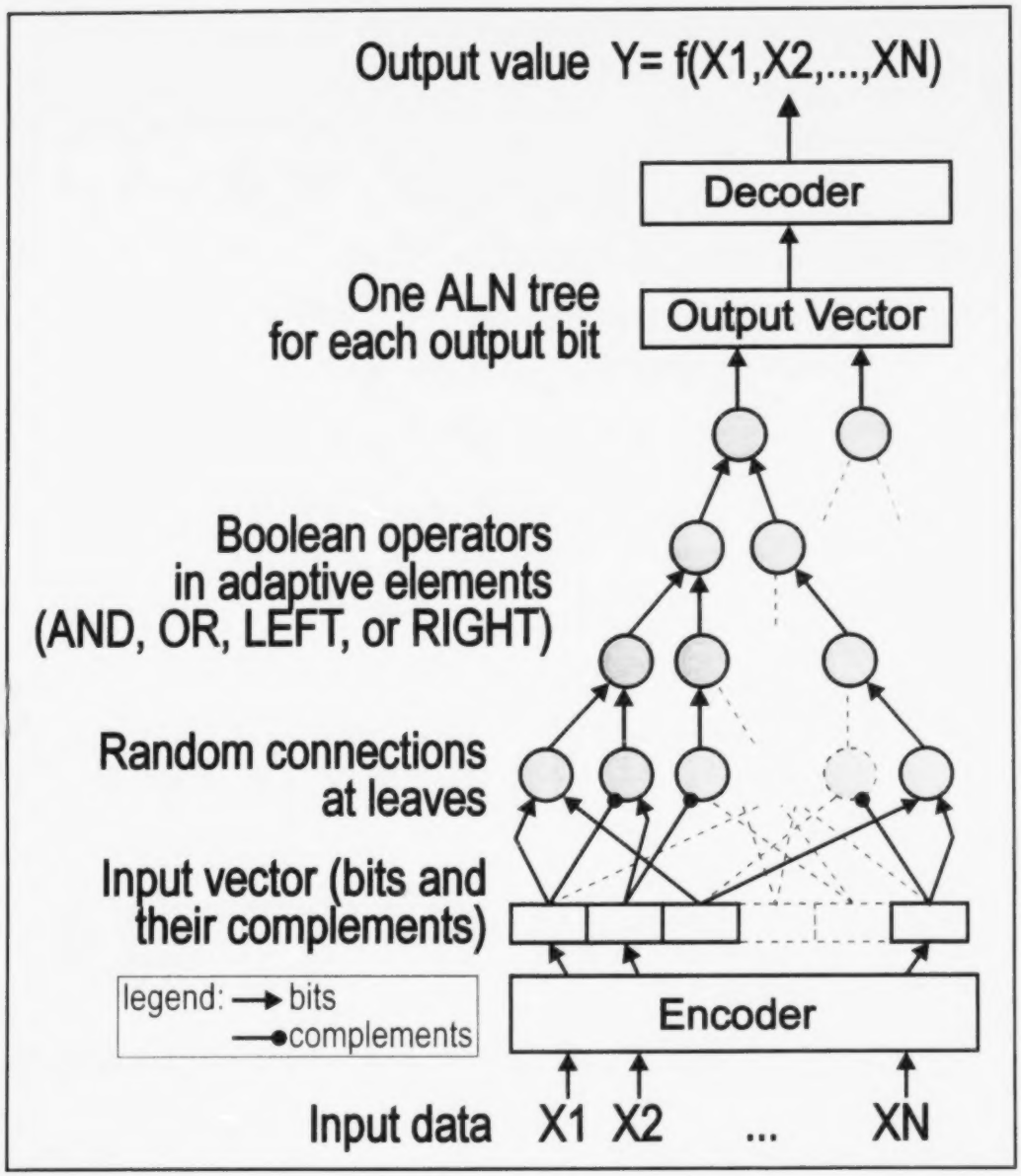

Figure 3.3.2. A structure of machine learning system based on adaptive logic networks (ALN version 2.x). Adaptation occurs in adaptive nodes where Boolean operators (AND, OR, LEFT and RIGHT) are initialy distributed randomly. They are changed from one to another during adaptation process according to algorithm described in Appendix B. ALNs 2.x operate only on binary numbers. Thus, real values have to be quantized and encoded before entering adaptation part of the algorithm. For the same reason, result obtained from ALNs has to be decoded. Binary complements are introduced to enable ALNs to approximate both monotonic increasing as well as monotonic decreasing input/output functions.

3.3.2 ALN application

To detect the subject's intention to take a step with her disabled leg, Interlink force sensing resistors (FSRs) were installed in her shoes to record the pressure under the medial metatarsal regions and under the heels. Although this type of sensor has a highly nonlinear force-resistance curve, and is not recommended for precise force measurement, it was acceptable for detecting ground contact. The complete list of inputs and outputs used in this design (including those calculated) is presented in Table 3.3.1 and the sample of input signals is shown in Figure 3.3.3.

Table 3.3.1. ALN inputs and outputs. The last two rows are calculated signals by the ALNs.

Signal Name	Role in ALN	Source of the signal
RIGHT FRONT	Domain	14.5 cm^2 square Force Sensor
RIGHT HEEL	Domain	4.9 cm^2 circle Force Sensor
LEFT FRONT	Domain	14.5 cm^2 square Force Sensor
LEFT HEEL	Domain	4.9 cm^2 circle Force Sensor
MANUAL SWITCH	Codomain	Manual switch
ALN	ALN Result	ALN prediction of Codomain
ERROR	ALN Error	= (MANUAL SWITCH - ALN)

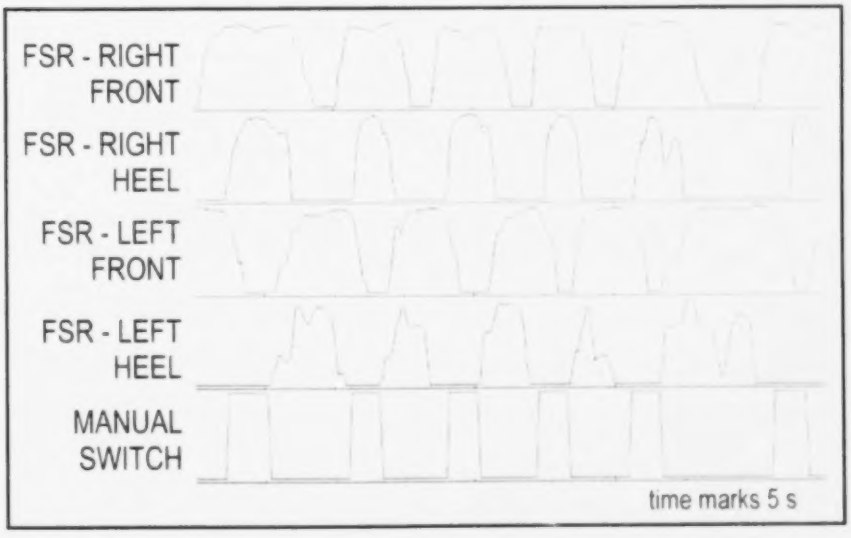

Figure 3.3.3. An example of the signals recorded from four force sensors and manual switch during FES-assisted walking controlled manually by the subject.

Since inputs and outputs of boolean functions are binary numbers, input signals have to be presented to the ALN program in binary form. If input and output data are analog signals, they are quantized and encoded before presentation to the learning algorithm. Encoding was done using a 'random walk technique' with a binary width of 10 or 14 bits for the inputs and 1 bit for the output. The number of quantization levels was 32 or 128 for the sensory inputs and 2 for the stimulation control signal. Note that the stimulation control is a binary signal to turn the stimulus on or off, produced by the subject or the therapist. The boolean trees were limited to 1024 or 4096 nodes, the number of voters was 5 and number of data presentations was 15. The ALN Result is the approximated stimulation control signal computed by the ALN, and ERROR is the difference between actual and the approximated control signal.

3.3.3 Results

Two sets of data were analyzed in which either the subject or the physiotherapist controlled the stimulation by pressing on the manual switch. The signals were recorded using the Axotape data acquisition system. The sampling rate of the analog input signals was 50 Hz, but it was reduced during processing to 5 Hz after low-pass filtering at 1 Hz. This was done to reduce the number of samples and the amount of computation during the training. In both sets of data the subject covered the same distance, turning around approximately in the middle. The data recorded from independent walking sessions were divided into training and test sets to evaluate the performance of the rules derived in recognizing when to 'press' the switch. Approximately half of the total number of samples was used for training the ALN, and the other half was used for evaluation. The ALN performance was measured by counting the number of correct samples, either those restored during training or those predicted during test.

3.3.3.1 Number of inputs

To measure the significance of each input signal for the ALN prediction, the use of all possible combinations of input signals from one and two sensors was evaluated when the physiotherapist controlled the stimulation. In all trials the ALN gave correct predictions for more than 65% of the time (the mean and standard deviations are shown in Figure 3.3.4. Only one combination of input signals was evaluated when the subject controlled the stimulation.

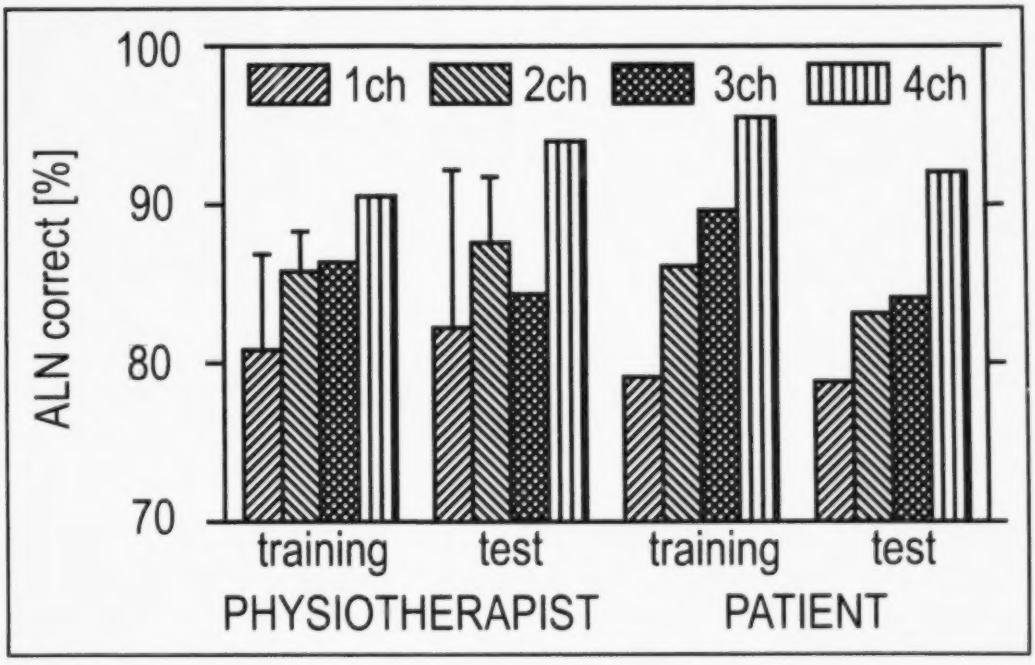

Figure 3.3.4. Adding more input signals can increase the number of correct predictions both for training and test.

3.3.3.2 The effect of past data

The impact of data from the past on predictions was investigated by using previous samples delayed in multiples of 0.4 s. In Figure 3.3.5, values at 1 represent the ALN results obtained using current sample of four input signals without previous samples. The values at two, three, four and

five represent the ALN results obtained using one, two, three and four previous samples of four input signals respectively, delayed 0.4 s each from previous one, in addition to current samples.

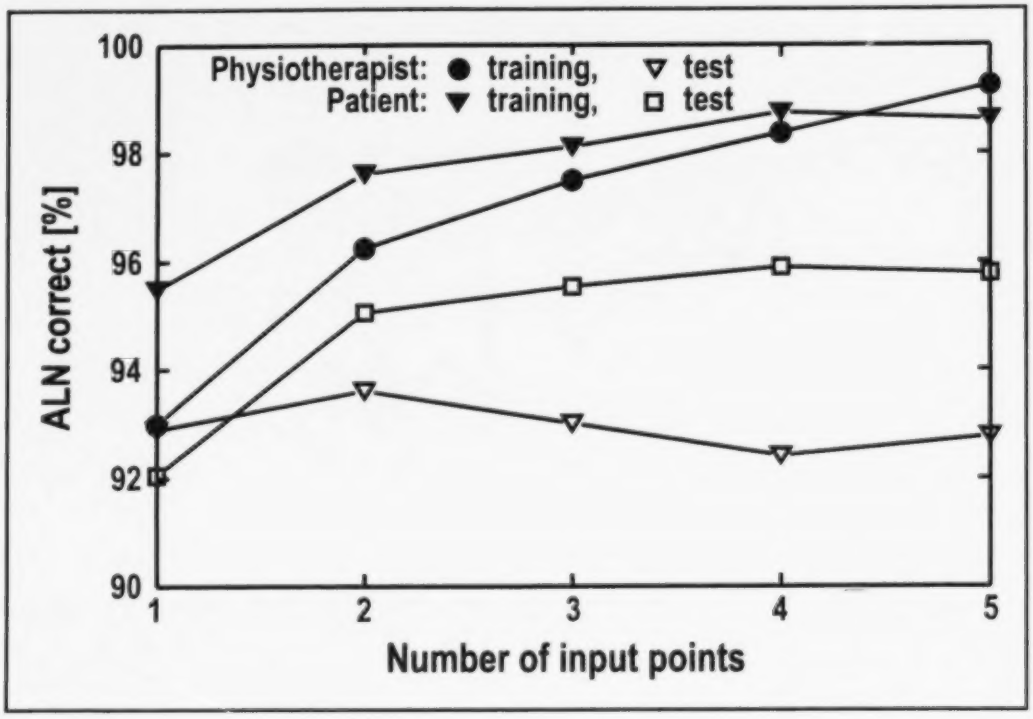

Figure 3.3.5. Adding past data can increase the number of correct responses during training, but may not during testing on new data, particularly when physiotherapist controlled stimulation. The number of samples was varied from only current one to five (current sample plus up to four previous samples at 0.4, 0.8, 1.2 and 1.6 s before the current sample).

3.3.3.3 The early prediction of the stimulation

The ability of the ALN to predict the future pattern of stimulation is demonstrated in Figure 3.3.6. Previous samples technique explained above was used to increase ALN performance. Two previous samples delayed 0.4 s each from previous one, in addition to the current sample of four signals (making 12 inputs) were used to predict state of the stimulation at the current time and up to two seconds in advance in increments of 0.4 s. Prediction was simulated by delaying input signals appropriately. Training error was almost constant, demonstrating ALNs capability to learn

training set independent from time shifts involved, however test error declined with increased prediction time still remaining above 84% at two seconds prediction.

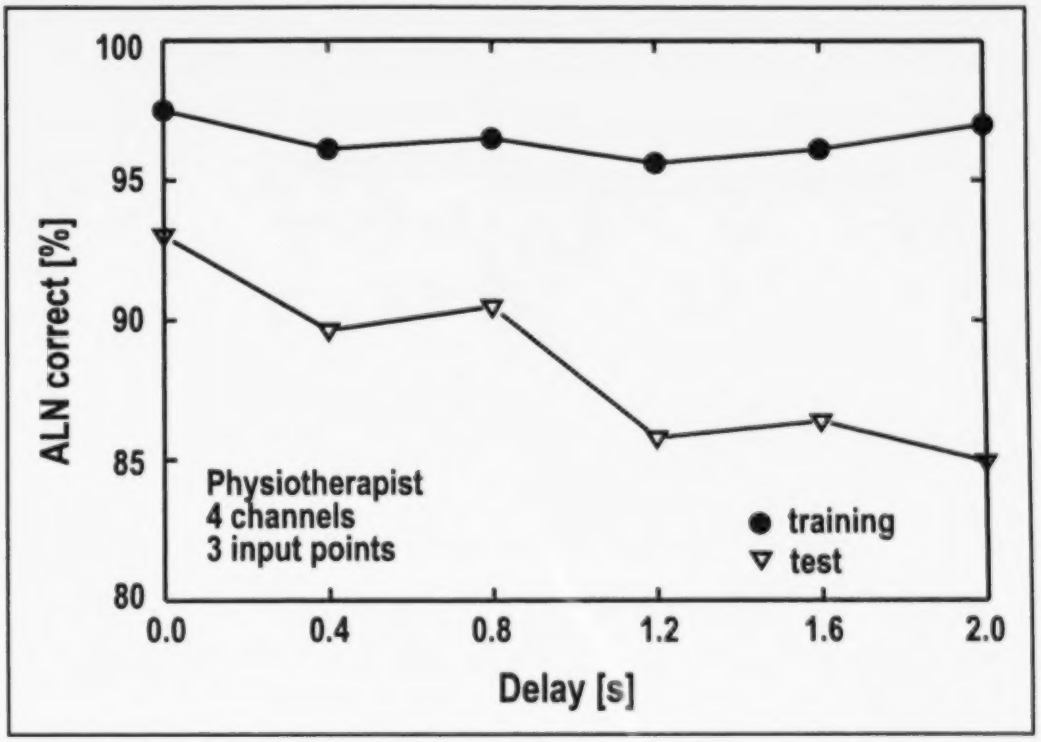

Figure 3.3.6. The ALN can predict future patterns of stimulation on the data used for training, but the number of correct responses declines when tested on new data.

3.3.4 Discusion and conclusion

Results of this evaluation show that ALN can be used for pattern recognition and control of simple FNS systems, if provided with enough information. Information coming from a particular input can be estimated by measuring the improvement or degradation of the training and prediction after adding or subtracting this input. The amount of information can be increased, either by using more inputs, or by using more past information from the same inputs. The best results are obtained by combining the two, as can be seen in Fig. 3.3.5. To attain 90% accuracy for

predictions required using at least four inputs. A good result in training does not mean that testing on previously unseen data will produce the same good prediction. This is referred to as 'overtrained trees' and it is important with neural nets to find optimal training parameters for a given type of input so that generalization gives the best results.

One interesting result was the ability of the ALN to predict stimulation events up to 2 s in the future. However, as shown in Fig. 3.3.6, generalization to the test data was degraded with increasing delay. An interesting possibility would be to predict the time of stimulation and inform the subject so that he or she could verify the prediction and, if correct, get prepared for the coming stimulus. If incorrect, a manual override could be used.

Two objectives are most important for future investigations of ALN: 1) improving the selection and preprocessing of inputs, and 2) optimizing the generalization to previously unseen data. The first objective probably does not depend on the ALN program, but the second one may be very dependent on encoding and training parameters of the ALN program. Its optimization can be approached by manipulations of input data, such as simulations of a range of walking speeds from a single speed of walking.

This program could be a very valuable tool in signal processing, pattern recognition and generating rules for control of gait for SCI subjects. However, the safety of its performance has not been verified under a wide range of conditions and until this has been accomplished the ALN must be combined either with algorithmic restrictions or expert systems that will prevent its unsafe operation.

3.4 EVALUATION OF IL FOR SWITCHING CONTROL OF FES[11]

Kirkwood and Andrews (1989) proposed inductive learning (IL) technique for deriving decision rules for gait event detection in an FES control system, based on a collection of examples of the events to be detected. The learning task was to learn the control strategy of a manually operated two-channel stimulation system for walking of a subject with incomplete spinal cord injury. Using ten attributes (input sensory signals), eight from force sensors built in shoe insoles and two measuring the load of crutches used to assist walking, they expected IL to mimic anticipatory actions that the subject performs while manually controlling two-channel stimulation. Their results have shown that IL is capable of learning such tasks with high accuracy and that it can identify redundancy in the sensory set used for training.

The study presented in the following section builds on the foundations set by the above mentioned reference and explores the feasibility of IL in more complex control design for the walking of subjects with complete spinal cord injury.

3.4.1 Methods

3.4.1.1 Subject

In this study we concentrated on a child (A.M. in Table 2.1.1) having a T2 level complete spinal cord injury and walking with long-leg braces (fixed knee). Stimulating her common peroneal nerves produces a flexion withdrawal reflex (i.e., flexion at the hip which was not braced) to assist

---

[11] A version of this section is accepted in form of short paper "Inductive Machine Learning in Control of FES-assisted Gait After Apinal Cord Injury" for presentation at the *5th Vienna International Workshop on Functional Electrical Stimulation*, Aug. 17-19, 1995, by Kostov et al. (1995b). Longer version of this section is in preparation for publication in form of paper "The use of inductive machine learning to select the most appropriate sensors to predict gait events: A case study in the application to automatic control of FES-assisted gait after spinal cord injury" by Kostov et al. (1995c).

in swing phase. Switching the stimulation ON and OFF for both legs was controlled by the subject manually pressing on the corresponding switches installed on the wheeled four-point walker. This operation required the use of both hands and a great cognitive effort. Reduction of the cognitive load by automating the switching control of the stimulation was the major motivating element for all participants in this study.

The stimulator used is a four channel Quadstim (Biomech Design Inc.), and the stimulation was applied through reusable conductive self-adhesive surface electrodes (Catalog No. 42004, Chattanooga Corporation, U.S.A.) positioned above the common peroneal nerves of both legs.

3.4.1.2 Recording setup and signal processing

Signals were recorded during four walking sessions from six circular (7/8" diameter) force sensors (FSRs) installed in the subject's shoes to measure the pressure under her feet and four inclinometers (Midori UV-1B) installed on her braces to measure inclination of her hips in two orthogonal planes, ie. hip flexion-extension (FE) and adduction-abduction (AA). Each of the walking sessions consisted of 20 to 24 steps and covered approximately five to eight meters in distance. The gain applied on signals recorded from FSRs and inclinometers was 1.1 and 2 respectively. Signals were recorded using a 12-bit Axotape data acquisition system (Axon Instruments, Inc.). The digital signals obtained were low-pass filtered (0.5 Hz) by a phase cancelled, fourth order Butterworth digital filter (Barr and Chan 1986). An example of the signals recorded and preprocessed as described above is presented in Figure 3.4.1. Trapezoidal pulses in the last two traces do not represent the actual duration of the stimulation, but only the intervals that the subject held the switch pressed. The duration of the stimulation on both channels was preset to 0.8 s.

Figure 3.4.1. Signals recorded from six FSRs measuring ground reaction force (GRF), two biaxial inclinometers, each measuring two hip joint angles, and two manual switches during FES-assisted walking controlled manually by the subject A.M. (next page)

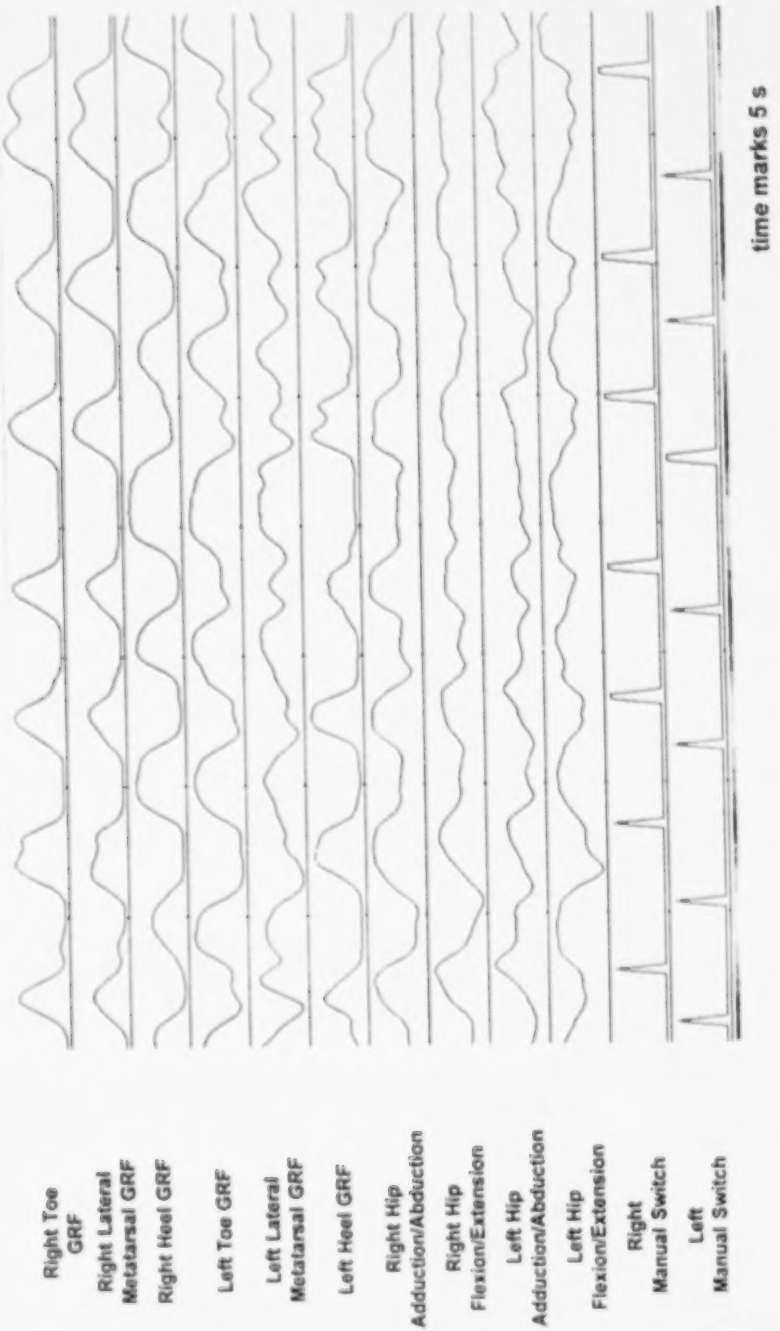

Figure 3.4.1. Signals recorded from six FSRs measuring ground reaction force (GRF), two biaxial inclinometers, each measuring two hip joint angles, and two manual switches during FES-assisted walking controlled manually by the subject A.M.

The critical characteristic of an inductive learning technique is the size of the decision tree. As the tree grows, most of the positive features listed in the section 2.7 and Appendix B become invalid because the learning algorithm, after extracting all important and invariant features from the input/output relationship, continues to learn details due to noise, which increases the training time and produces large incomprehensible trees despite the fact that they are still explicit. The training for IL is usually very fast (within 10 seconds on an IBM PC 486 DX/50 MHz machine for 1000 samples) and it was optimized with regard to preset training error (Kostov et al., 1995). The actual program used for implementation of the inductive learning technique, EMPIRIC, was provided by Heller (1992). The program was originally written in the Pascal programming language and its algorithm was not optimized for speed of execution.

In the first step of automatic design of the rules for the control system, the inductive learning algorithm was used to reduce the number of sensors and, consequently, the complexity of the control system. After the set of sensors was reduced by excluding those of least importance for a particular decision, the derived signals and past data samples were used together with the original signals to intuitively improve rule induction. The quality of the rule induction was quantified by its ability to generalize and this was estimated by counting wrongly predicted samples in data sets not used for training. The number of such samples represents the test error and it is expressed as a percentage of the total number of samples in a given data set.

After the decision tree was created, it was tested on the other three data sets. Figure 3.4.2 shows predictions from the decision tree (dashed line) and actual stimulation intervals (solid line) initiated by the closure of the right hand switch by the subject. In this example, the percentage of incorrectly predicted samples (generalization error) is 4.1%. The functional error is much smaller than this number, because the stimulation could start earlier or later than they actually did without significantly affecting the walking.

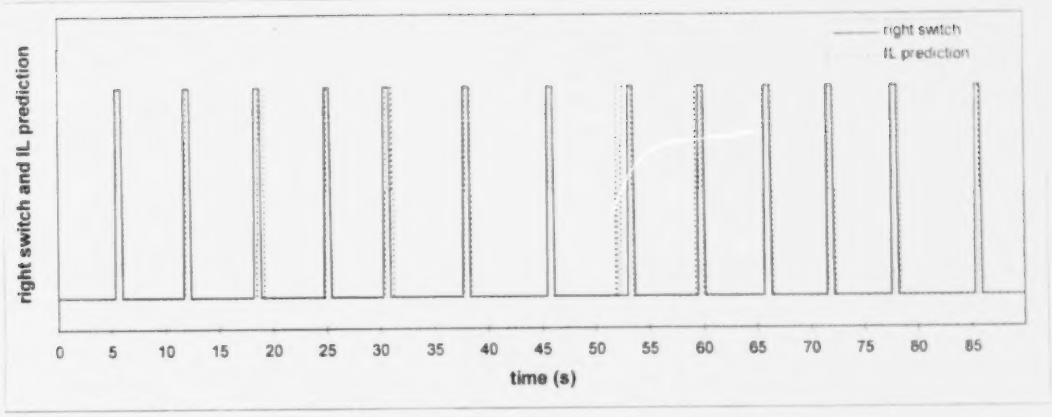

Figure 3.4.2. An example of IL prediction (dashed line) of actual stimulation intervals initiated by right hand switch closure (solid line) during FES-assisted walking.

3.4.2 Results

3.4.2.1 Reduction of the number of sensors

Estimating the relative importance of sensors: Each training operation on one of four basic data sets resulted in a decision tree similar to that in Figure 3.4.3. Importance of attributes used in training was estimated using the number of samples classified at particular level of decision tree. Depending on the relative position of the first occurrence of the given signal in the decision tree, that signal's importance was estimated on a scale starting from one for the least important sensor (the last one to be used in decision tree) and ending with M for the most important of M sensor (the first one used at the top of the decision tree). The array of numbers representing the relative importance of sensors used in that particular decision tree is marked in bold in Figure 3.4.3.

The numbers were then averaged over all four data sets in predicting the right and left hand switches separately. Average numbers obtained are plotted for each sensor, as shown in Figure 3.4.4

number of nodes = 59 **RELATIVE IMPORTANCE OF SENSOR**

```
Node  1: IF (att  7 (R_HIP_AA  ) <= 288.50) THEN Node  2 ELSE Node  3   ..... 10 R_HIP_AA
Node  2: IF (att  7 (R_HIP_AA  ) <= 287.50) THEN Node 14 ELSE Node 15
Node  3: IF (att  8 (R_HIP_FE  ) <= 163.50) THEN Node  4 ELSE Node  5   .....  9 R_HIP_FE
Node  4: CLASS  1 (STIM_ON     ) No. Examples correct 69 incorrect  0
Node  5: IF (att  4 (L_TOE     ) <=  92.50) THEN Node  6 ELSE Node  7   .....  8 L_TOE
Node  6: IF (att  8 (R_HIP_FE  ) <= 209.00) THEN Node  8 ELSE Node  9
Node  7: IF (att 10 (L_HIP_FE  ) <=  61.50) THEN Node 10 ELSE Node 11   .....  7 L_HIP_FE
Node  8: CLASS  1 (STIM_ON     ) No. Examples correct 42 incorrect  0
Node  9: CLASS  2 (STIM_OFF    ) No. Examples correct  4 incorrect  0
Node 10: IF (att  9 (L_HIP_AA  ) <= -193.00) THEN Node 16 ELSE Node 17 .....  6 L_HIP_AA
Node 11: IF (att  5 (L_LAT_MET ) <= 280.50) THEN Node 12 ELSE Node 13 .....  5 L_LAT_MET
Node 12: CLASS  1 (STIM_ON     ) No. Examples correct 26 incorrect  0
Node 13: IF (att  3 (R_HEEL    ) <= 155.50) THEN Node 34 ELSE Node 35   .....  4 R_HEEL
Node 14: CLASS  1 (STIM_ON     ) No. Examples correct635 incorrect  0
Node 15: IF (att  1 (R_TOE     ) <= 330.00) THEN Node 58 ELSE Node 59   .....  3 R_TOE
Node 16: IF (att  1 (R_TOE     ) <= 393.00) THEN Node 18 ELSE Node 19
Node 17: IF (att  5 (L_LAT_MET ) <= 235.50) THEN Node 32 ELSE Node 33
Node 18: IF (att  3 (R_HEEL    ) <=  74.50) THEN Node 38 ELSE Node 39
Node 19: IF (att  1 (R_TOE     ) <= 532.00) THEN Node 20 ELSE Node 21
Node 20: IF (att  9 (L_HIP_AA  ) <= -222.50) THEN Node 22 ELSE Node 23
Node 21: IF (att  6 (L_HEEL    ) <= 517.00) THEN Node 28 ELSE Node 29   .....  2 L_HEEL
Node 22: IF (att  2 (R_LAT_MET ) <= 410.50) THEN Node 26 ELSE Node 27 .....  1 R_LAT_MET
Node 23: IF (att  3 (R_HEEL    ) <=  66.00) THEN Node 24 ELSE Node 25
Node 24: CLASS  1 (STIM_ON     ) No. Examples correct  5 incorrect  0
Node 25: CLASS  2 (STIM_OFF    ) No. Examples correct  3 incorrect  0
Node 26: CLASS  2 (STIM_OFF    ) No. Examples correct 27 incorrect  0
Node 27: CLASS  1 (STIM_ON     ) No. Examples correct  1 incorrect  0
Node 28: IF (att 10 (L_HIP_FE  ) <=  55.00) THEN Node 30 ELSE Node 31
Node 29: CLASS  2 (STIM_OFF    ) No. Examples correct  4 incorrect  0
Node 30: IF (att  1 (R_TOE     ) <= 571.00) THEN Node 40 ELSE Node 41
Node 31: CLASS  2 (STIM_OFF    ) No. Examples correct  4 incorrect  0
Node 32: CLASS  1 (STIM_ON     ) No. Examples correct  1 incorrect  0
Node 33: CLASS  2 (STIM_OFF    ) No. Examples correct 14 incorrect  0
Node 34: IF (att  7 (R_HIP_AA  ) <= 351.50) THEN Node 36 ELSE Node 37
Node 35: CLASS  2 (STIM_OFF    ) No. Examples correct  5 incorrect  0
Node 36: CLASS  1 (STIM_ON     ) No. Examples correct  3 incorrect  0
Node 37: CLASS  2 (STIM_OFF    ) No. Examples correct  2 incorrect  0
Node 38: IF (att 10 (L_HIP_FE  ) <= -20.50) THEN Node 52 ELSE Node 53
Node 39: CLASS  1 (STIM_ON     ) No. Examples correct  9 incorrect  0
Node 40: IF (att  1 (R_TOE     ) <= 552.50) THEN Node 42 ELSE Node 43
Node 41: CLASS  2 (STIM_OFF    ) No. Examples correct  2 incorrect  0
Node 42: IF (att  5 (L_LAT_MET ) <= 149.50) THEN Node 44 ELSE Node 45
Node 43: IF (att  3 (R_HEEL    ) <=  57.50) THEN Node 46 ELSE Node 47
Node 44: IF (att  2 (R_LAT_MET ) <= 373.00) THEN Node 50 ELSE Node 51
Node 45: IF (att  9 (L_HIP_AA  ) <= -244.00) THEN Node 48 ELSE Node 49
Node 46: CLASS  1 (STIM_ON     ) No. Examples correct 17 incorrect  0
Node 47: CLASS  2 (STIM_OFF    ) No. Examples correct  1 incorrect  0
Node 48: CLASS  2 (STIM_OFF    ) No. Examples correct  1 incorrect  0
Node 49: CLASS  1 (STIM_ON     ) No. Examples correct  7 incorrect  0
Node 50: CLASS  2 (STIM_OFF    ) No. Examples correct  6 incorrect  0
Node 51: IF (att  1 (R_TOE     ) <= 545.50) THEN Node 56 ELSE Node 57
Node 52: CLASS  1 (STIM_ON     ) No. Examples correct  3 incorrect  0
Node 53: IF (att 10 (L_HIP_FE  ) <=   6.00) THEN Node 54 ELSE Node 55
Node 54: CLASS  2 (STIM_OFF    ) No. Examples correct  3 incorrect  0
Node 55: CLASS  1 (STIM_ON     ) No. Examples correct  2 incorrect  0
Node 56: CLASS  2 (STIM_OFF    ) No. Examples correct  1 incorrect  0
Node 57: CLASS  1 (STIM_ON     ) No. Examples correct  2 incorrect  0
Node 58: CLASS  1 (STIM_ON     ) No. Examples correct  1 incorrect  0
Node 59: CLASS  2 (STIM_OFF    ) No. Examples correct  1 incorrect  0
```

Figure 3.4.3. Text form of IL decision tree showing IF(...) THEN(...) ELSE(...) rules and the extracted signal importance.

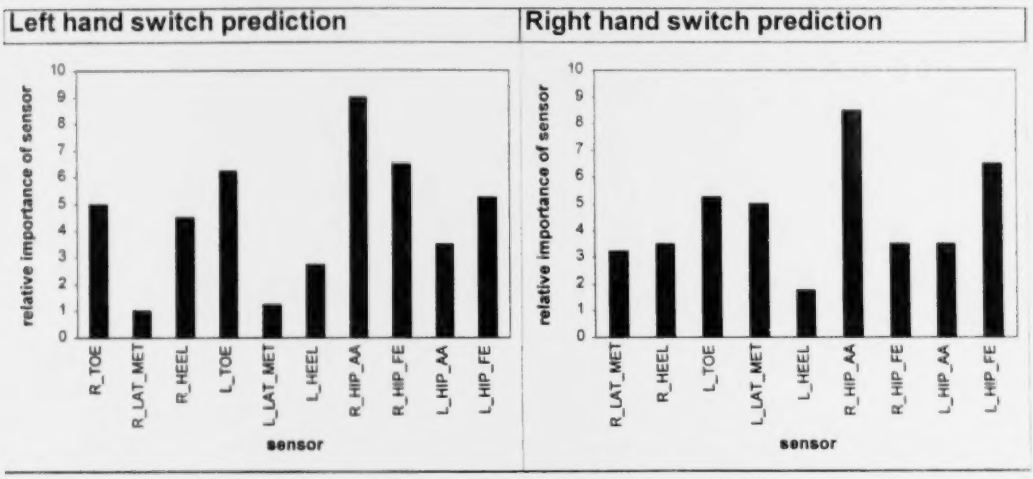

Figure 3.4.4. The average importance of sensors used in IL training. Higher the bar in the graph, more important corresponding sensor is for this particular task.

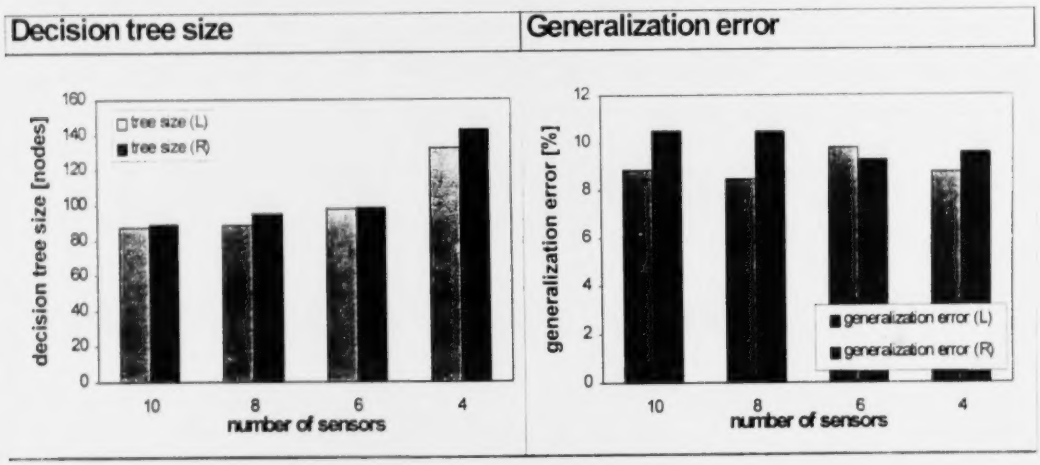

Figure 3.4.5. Reduction of the number of sensors by inductive learning.

Reduction in the number of sensors: The number of sensors was reduced from ten to four in three steps. In each step the two least important sensors i.e. the sensors with lowest values were excluded. The trade-off for this reduction of the decision tree size was a significant increase in the

average size of the decision tree (from 88 and 89 to 133 and 143 nodes for left and right hand switch prediction respectively). The largest decision tree size increase occurrs when the number of sensors is reduced from six to four, suggesting that four sensors carries significantly less information than set of six sensors.The generalization error remained approximately in the same range between 9.5 and 10.5 % as for decision trees made by using all ten sensors, which suggests a high redundancy in the sensory information supplied to the learning algorithm.

Summary of these results is shown in Figure 3.4.5.

3.4.2.2 Optimization of the training using reduced set of sensors

The size of the decision trees and generalization of the learning method on the reduced set of sensors can be improved by using the present signal values together with previous samples, thus adding a memory (or time dimension) to the method (Veltink, 1990; Kostov et al., 1992).

Use of past data points: The inductive learning algorithm has only spatial characteristics, which means that the mapping of the input/output relationship is achieved in a multidimensional input/output space excluding the time dimension. In its original form the algorithm does not use any temporal characteristics of the input/output relationship. To add the time dimension to the mapping algorithm, memory or signal samples from the past can be used in addition to current ones in predicting the outputs for the current sample. To illustrate this technology, up to the three previous samples in addition to the current sample of all original four signals recorded from the reduced set of sensors were used for training of the current stimulation outputs. The use of previous samples decreased the average decision tree size below the corresponding size obtained using ten sensors. It has also significantly decreased the test error (see Figure 3.4.6).

Use of a differentiated signals: Further improvement of the rule induction was achieved using both the original and differentiated signals and, in the most successful of all training trials, with one sample from the past. The use of more than one sample from the past would be even more

productive, but it was restricted by the program implemented which could not deal with more than 16 input signals. Both the size of decision tree and the generalization error achieved were below those resulting from the experiments with all ten sensors (see Figure 3.4.7).

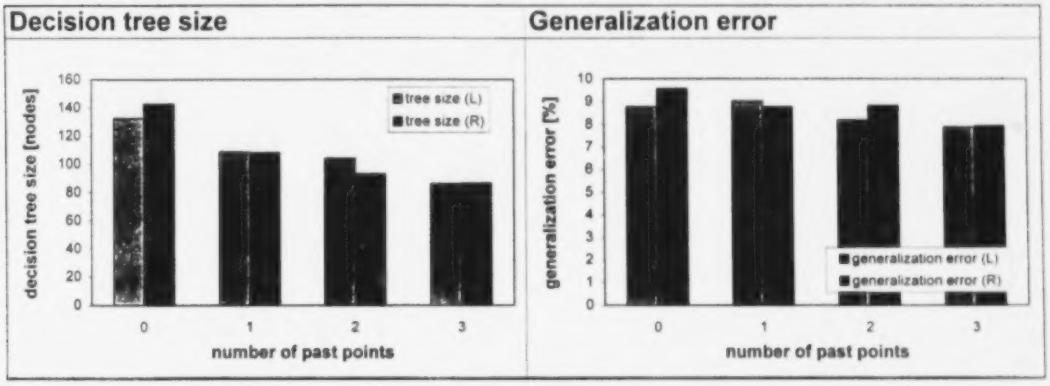

Figure 3.4.6. The use of previous signal samples in addition to the current ones significantly reduces the size of the decision tree and the generalization error.

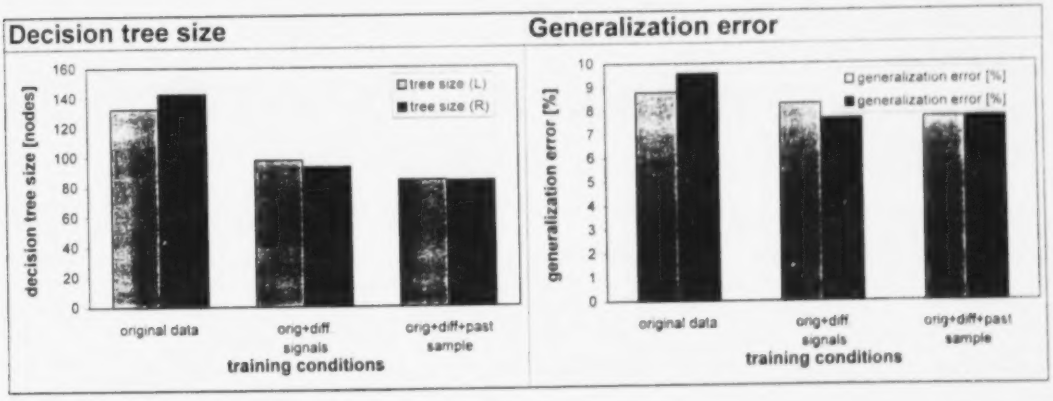

Figure 3.4.7. The use of a differentiated signal and one previous sample in addition to the original four sensory signals resulted in the smallest decision trees and generalization error.

3.4.3 Discussion and conclusion

This study demonstrated the potential of inductive learning technique for use as a powerful tool in designing control systems for FES-assisted locomotion of subjects having a complete spinal cord injury. IL was used to map the functional relationship between sensory feedback signals and manually produced stimulation control signals and to create control rules which can replace manual control. Sensory feedback signals were recorded from six force sensing resistors, installed in the subject's shoes to measure ground reaction forces under her feet, and four inclinometers (tilt sensors) installed on her braces to measure inclination of her hips in two orthogonal planes, i.e. hip flexion-extension and adduction-abduction. Stimulation control signals were produced by the subject pressing on two switches installed on the handles of her walker.

Kirkwood and Andrews (1989) reported the use of IL in measuring the significance of sensory input for particular mapping. The same feature was used in this experiment to reduce the number of sensory inputs from initial ten to only four. Reduction of number of sensors increased slightly the size of the decision trees but it did not change IL test error. This result demonstrates high redundancy in the system. The size of decision trees was brought down to the initial level by optimizing the learning setup with reduced set of sensors. Past data samples and differentiated signals were used together with the original four signals to obtain smaller decision trees and even lower test errors than with all ten sensors.

3.5 ALN - IL COMPARISON[12]

After obtaining encouraging results in the initial feasibility studies with both adaptive logic networks and inductive learning, the learning and generalization performance of both techniques were compared on different control tasks for FES-assisted walking of several subjects. Some of the intrinsic features of both techniques were already known, but from this study we expected to obtain measurements of performance that would affect the final implementation of a control system for FES-assisted locomotion of spinal cord injured subjects.

3.5.1 Methods

Version of the ALN learning program: **ALN V.2** (see section 2.7.1 and Appendix B for details).

ALN interface program: **WALKON** (see section 2.8.1 and Appendix C for details).

Encoding technique: **Unary Encoding** (see section 2.7.1.1).

3.5.1.1 Subjects

We concentrated on subjects with incomplete spinal cord injury that limited their ability to ambulate. All of the subjects, except L.W. were able to stand and walk with stimulation for a short distance. The data used in the present study were recorded from the first six SCI subjects presented in Table 2.1.1, during routine gait analysis, and stored for off-line processing. The information about their injuries, level of disability and current rehabilitation status related to the use of FES is summarized in Table 2.1.1. All subjects signed the consent form approved by a local ethics committee.

---

[12] A version of this section has been accepted for publication in IEEE Transactions on Biomedical Engineering, in form of paper "Machine learning in control of functional electrical stimulation systems for locomotion" by Kostov et al. (1995a).

3.5.1.2 Methods of recording

The signals were recorded from two types of transducers: force sensing resistors (Interlink Electronics, 1990a) positioned under the medial metatarsal and heel areas of both feet in all subjects, and flexible goniometers (Penny and Giles, 1994). Goniometers have been attached across the joints of the affected leg and used in four subjects to measure the hip, knee and ankle joint angles. The subjects controlled the FES system by using either a finger-operated switch mounted on the hand grip of a walking aid (walker, crutch or cane), or a heel-switch on the insole of the subject's shoe (see FES control column in Table 2.1.1). The switching events were also recorded. An example of signals used in the experiment is shown in Figure 3.5.1.

A particular recording setup was decided for each subject, depending upon the severity of injury, amount and types of bracing, stimulation necessary for locomotion, walking aids and the subject's overall condition at the time of recording. The subjects were instructed to walk in one direction without unnecessary stops for six to eight meters, and, if possible, to turn around and walk back to the starting position. In one subject (L.W.) we recorded three 30 m sequences of straight walking outside the gait laboratory. The experimentally determined stimulation parameters were used to produce a gait that was as steady as possible with the least psychological and physical fatigue for the subject. At the beginning of each recording session there was a setup phase used to adjust stimulation intensity, duration of the stimulation period and the position of the electrodes (or RF transmitting antenna in the case of the subject L.W. with a fully implanted stimulator, modified Mikrofes, Ljubljana). After satisfactorily adjusting parameters one or more walking rounds were completed and the data were recorded for analysis.

Signals from transducers were amplified, low-pass filtered (25 Hz), sampled at 50 Hz by a 12 bit A/D converter (Axon Instruments, Inc.) and recorded in a digital form using an IBM PC compatible computer. Low-pass filtering was used mainly to avoid high frequency noise, if there was any. The useful signal, whose power was concentrated far below 5 Hz was not affected by this filtering.

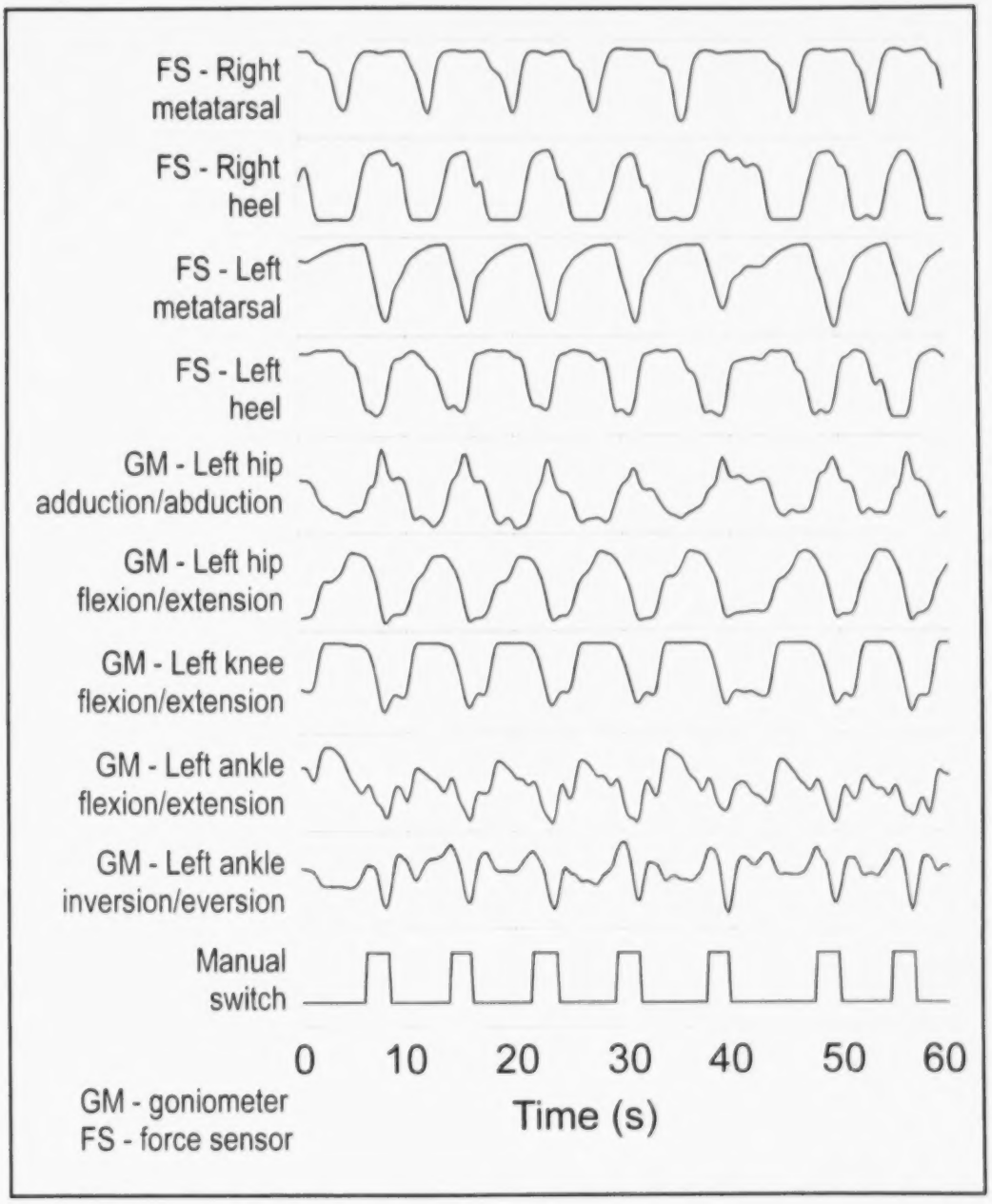

Figure 3.5.1. An example of training signals recorded in subject L.W. during FES-assisted walking. Sensors installed in subject's shoes (FS) measure ground reaction force in N, and goniometers (GM) attached across the joints of the stimulated leg measure joint angle variations in deg. Stimulation was controlled by the subject manually pressing on the switch (bottom trace). Multi-channel inputs are required for safe control but they make analysis of the recorded data difficult.

Signals recorded from six subjects were organized in forty-eight data files, which consisted of one to nine inputs (signals from the transducers) and one or two outputs. These outputs are the signals from the switching devices (e.g., manual switch, heel switch), each used to control one or more FES channels. Every data file was split into halves marked A and B. Both halves were used as a training set in one experiment and a test set in another. The three largest files recorded from subject L.W. were split into thirds and used in the 'retraining experiment' which is described later.

3.5.2 Results

3.5.2.1 The overall performance of both techniques

The performance of the learning algorithms was tested by measuring the training and test errors. Average training and test errors and standard deviations for all ALN and IL training sessions are presented in Table 3.5.1. ALN training sessions were done using 2048 leaves and 2048 adaptive elements, a maximum of 20 presentations of the inputs to the learning algorithm, seven ALN trees per bit in the output vector and maximum training error of 6% (see Figure 3.5.2). IL training sessions were done using the same maximum training error of 6% (see Figure 3.5.3). Overall performance using only original data samples was similar for both algorithms. ALNs showed 7% smaller test error ((1-10.5/11.3)*100). At the same time, IL had 43% better training error ((1-3.3/5.8)*100) which is not very useful in applications where the training set represents only a sample in the multidimensional input space.

Table 3.5.1. Average training and test errors for all ALN and IL learning sessions.

	Adaptive Logic Networks		Inductive Learning	
	Training error [%]	Test error [%]	Training error [%]	Test error [%]
mean	5.8	10.5	3.3	11.3
SD	5.8	8.9	3.2	8.9

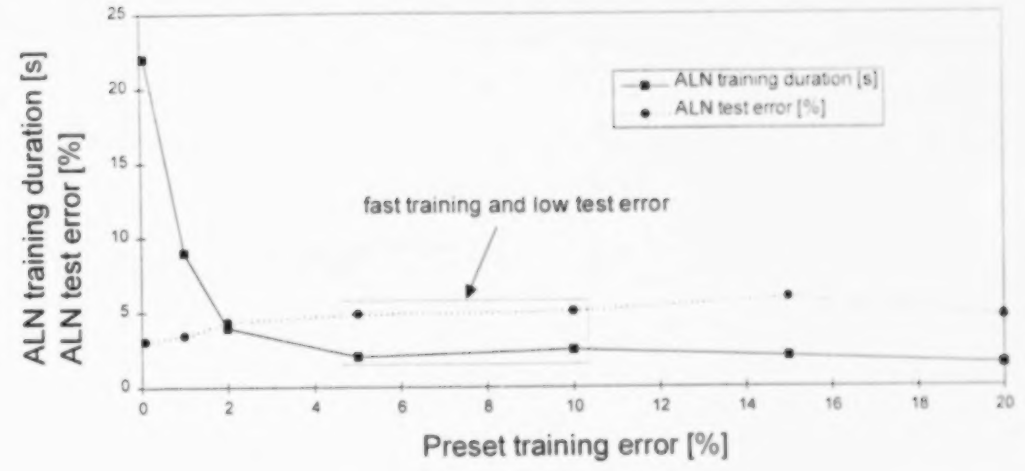

Figure 3.5.2. Optimization of ALN training with regard to training duration and test error.

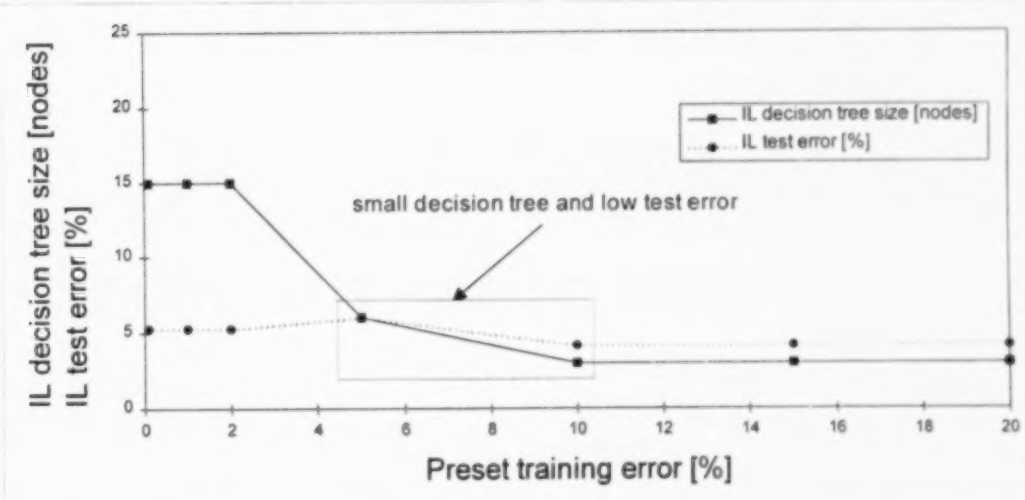

Figure 3.5.3. Optimization of the IL decision tree with regard to its size (number of nodes) and test error.

3.5.2.2 The use of previous samples in the training

The training could be improved significantly by the use of samples from the past, in addition to current ones. This brings a time dimension to the learning algorithm, which is very important for time-varying functions such as time-dependent signals recorded from artificial or natural sensors. The dependence of ALN and IL performance on the past sample delay in the range between 0 s (no sample from the past) and 2 s is tested. An example of the effects of a single sample from the past, and the delay between that sample and the current one, on the test error for both ALN and IL is shown in Figure 3.5.4. In this example the test error achieved using one sample from the past whose delay varied from 0.2 to 2 s was on average 28% smaller for the ALN than for the IL. This study showed that adding previous samples with properly chosen delays to the current samples, may reduce the test error for both algorithms.

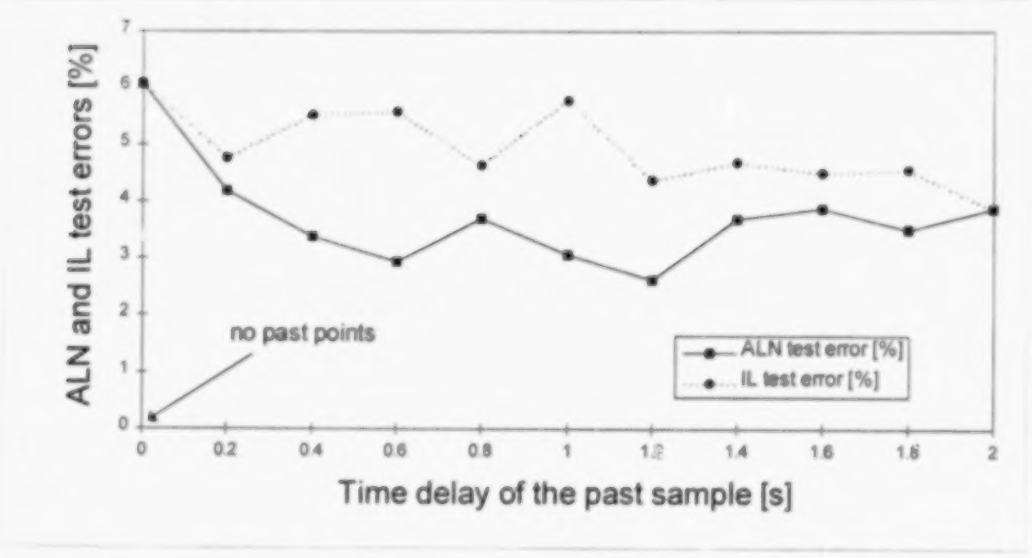

Figure 3.5.4. The use of a single sample from the past in addition to the current one reduces the test error. The delay of the sample from the past should be optimized for a particular data set. It is recommended to use more than one sample from the past if computational power allows it.

3.5.2.3 *Prediction of future stimulation events*

For control purposes it is very important to recognize the subject's intention to perform a certain action in advance (e.g. intention to take a step). The recognized action may then be technically verified and announced to the subject for his or her acceptance or rejection. Two seconds was considered a sufficient period of time for this process, and both algorithms are able to provide the necessary prediction. An example of the effects of different intervals used in early prediction on the test errors obtained by the ALN and the IL is shown in Figure 3.5.5. In this example the test error achieved while predicting future stimulation from 0.2 to 2s in advance was on average 17% smaller and the standard deviation of the error was 31% smaller for the ALN than for the IL. The ALN usually showed smaller test error and less variability of the results within the range examined.

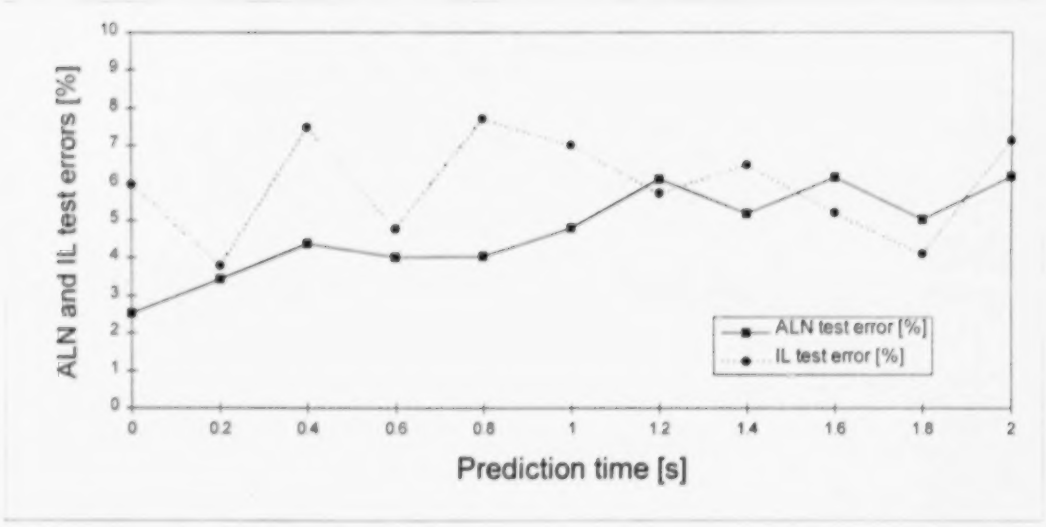

Figure 3.5.5. It is possible to predict future stimulation events by both techniques. There is a simple rule in event prediction by the ALN: the longer the prediction time, the bigger the test error. Because of larger variability in IL results, the above rule does not always apply to IL. This example shows that a smaller error and a smaller variability was achieved by the ALN.

3.5.2.4 Characteristics that are unique to one of techniques

Some of the features were impossible to compare because they relate to only one of the techniques. An important feature of an ALN is a tree ***retraining*** capability. This capability provides repetitive adjustment of existing trees to new data without significant degradation of the knowledge stored in them by previous training. The following experiment was designed to demonstrate the ALN's capability to retrain existing trees. Performance of the retrained ALN trees was compared to that produced by regular IL training.

Three data files were divided into thirds: A, B and C. In each test for both ALN and IL the training was done, as usual, on one of the parts and performance was measured by evaluation on the same (training error) and other parts of the same data file (test error). These results are shown and compared in parts 1 and 2 of the graph in Figure 3.5.6 respectively. There is a slightly smaller error achieved by IL compared to the one achieved by the ALN during training but a slightly smaller test error for the ALN. In the next phase of the experiment ALN trees were retrained with the data from the next consecutive part of the same data file. The test was repeated on the last part of the same data file and the test error achieved is presented in part 3 of the Figure 3.5.6. It shows significant reduction of the test error after retraining the existing trees (from 15.9% to 8.1% on average). The last phase of the experiment was to demonstrate that the loss of the stored knowledge in previously trained trees is not significant after retraining. After the retraining on the second part of the data file, the test was done on the first part of the same data file used for training. The resulting error is shown in part 4 of Figure 3.5.6 and it should be compared with the ALN bar in part 1 of the same figure. The error increased from 5% to 6.4% on average.

Retraining of the IL trees has not been reported to date. It is possible to store samples from previous experiments and to use them together with the new samples in retraining, but this is limited by the amount of available memory. In addition, for large number of samples the time required for training may also become the limiting factor.

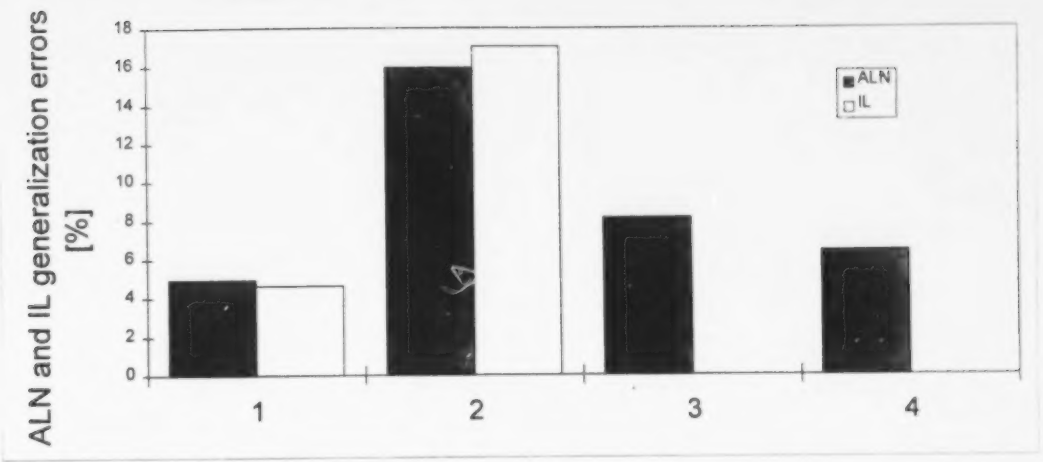

Figure 3.5.6. Adaptive logic in ALNs provides the user with a retraining capability. In this graph ALN retraining was compared to IL regular training and plotted as a reduction of the error resulting from retraining. The graph has 4 parts: 1) average training error for both ALN and IL techniques; 2) average test error for both ALN and IL techniques; 3) ALN test error after retraining; 4) test error on the first part of the data file after retraining (compare with ALN result shown in part 1).

The IL technique, on the other hand, produces small decision trees whose ***reasoning can be understood*** since the rules are of the familiar *IF(...) THEN(...) ELSE(...)* type. The IL technique furthermore supplies the ***information of relative importance*** (in terms of information theory) of the sensory data used. The order in which the input channels are used in a decision tree represents their relative importance; i.e., the closer the use of the input is to the top of the decision tree, the more important that input is for that particular learning task. Detailed presentation of this feature is shown in Chapter 3.4. It may be very useful in reducing the number of sensors during control system designing and customization to a particular task or a particular subject.

3.5.3 Discussion

The performance of two machine learning techniques (adaptive logic networks and inductive learning) was compared in replacing part of the voluntary control and the upper level controller in

simple FES systems. These systems are used by spinal cord injured subjects with incomplete lesions for assistance during ambulation.

An overall average test error is just seven percent smaller for ALN than for IL. This suggests that the quality of the learning algorithm does not depend as much on the technique used in training, as on the amount of information that selected sensory inputs carry into the training. Of course, this is valid only if the learning algorithm is optimized for that particular data set.

To bring the time dimension into the learning algorithm, we have tested the use of data samples from the past together with the current samples. The performance of both techniques was improved, and again the ALN showed a smaller error than the IL. This is probably due to structural differences in the two learning algorithms. The ALN uses all of its inputs at any time during the adaptation, while the IL uses only the input which gives the biggest mutual information gain at a particular partitioning step.

To verify necessary actions predicted by the learning technique and to signal the next actions to the subject, so that he or she may accept or cancel them, it is important to predict them in advance. We have tested both techniques in prediction of stimulation events by up to two seconds. Both techniques are capable of satisfying the learning task, and yet again the ALN technique showed a smaller test error and less variability than the IL technique. In addition, both algorithms showed the expected increase in test error with an increase of the prediction interval.

Besides the characteristics that are common to both algorithms, those which are unique to only one were evaluated. The most advantageous characteristic of the ALN is that trained trees may be retrained using new data without significant loss of previously collected knowledge. This is particularly important for combined man-machine control processes because during the adaptation period, both man and machine should be able to preserve already collected skill (or knowledge) and to modify it by new data. Bearing in mind that subjects should be able to put on different shoes, which may have a significant influence on recorded signals, and still use

previously learned functions, the retraining feature of the ALN will allow them to further train the decision trees without passing through the whole training process again. Advantages of the IL are fast training, small and comprehensible decision trees when there is enough information in input data (which has to be determined empirically), and provision of information about the relative importance of a particular input for a particular learning task. This information may be very useful when designing a control for an FES-system since the importance of the sensors in the detection of the event is not intuitively obvious (Kirkwood and Andrews, 1989). Therefore, information on relative importance would enable the designer to choose the most important sensors and perhaps to add additional ones for fault-tolerant applications.

A summary of the comparison between ALN and IL is presented in Table 3.5.2.

Table 3.5.2. A summary of the comparison between ALN and IL.

	ALN V.2	**IL**
training speed	high (seconds)	high (seconds)
generalization	very good	good
type of decision trees	binary	IF () THEN () ELSE
size of the trees [nodes]	>1000	< 20
past points use	easy	not easy
future events prediction	yes	yes
retraining	yes	no
comprehensible trees	no	yes
input significance information	not available	available
hardware implementation	microproc. + FPGA	microproc.

3.5.4 Conclusion

The results suggest that both algorithms can be used successfully in this application, but they may require specific fine tuning of the factors that influence their generalization on new data, such as the size of the training set and the preset training error level. Neither approach is better in all circumstances. Most tasks can be done with either algorithm, although the amount of work required to do the task may differ. Features that are unique to only one technique may be the

determining factor in deciding which technique to employ in a particular design. If the complexity of design will benefit from the advantageous features of ALN and if a retraining feature is required, then adaptive logic networks are the choice. If small, understandable decision trees are preferred, then the choice is inductive learning technique. In a future work we plan to combine both techniques in order to exploit all of their advantages. Inductive learning technique may be very useful for estimation of the relative importance of sensory inputs for particular design, and ALN may be used in a final control system for its better generalization and retraining feature.

3.6 REAL-TIME SWITCHING CONTROL OF FES BY ALN[13]

After the evaluation of two different machine learning techniques (MLTs) demonstrated the potential of these techniques for automatic rule-generation for control of FES-assisted walking, an integrated control system (ICS) was designed to implement the same approach in real-time walking. This approach is presented in great detail in earlier sections, but it is briefly introduced here again. In the automation of stimulation control for FES-assisted walking, the MLT is used to extract invariant characteristics of the relationship between feedback sensory signals and stimulation control signals, to store these characteristics in the form of decision trees, and then to use these decision trees together with feedback sensory signals for prediction of stimulation control signals. This experiment goes far beyond the feasibility and evaluation studies presented in earlier sections of this chapter. The decision trees formed during the training are applied to real-time control of walking by a subject with incomplete spinal cord injury. The major difference between the evaluation experiments and this experiment is that real-time control can not be simulated and tested off-line. This increased the physical and psychological stress in the subject. To protect the subject in hazardous situations, the actions proposed by the MLT were filtered through hand-crafted restriction rules limiting stimulation parameters only to values belonging to the range measured in the training data set. Additional safety measures for the subject were introduced to prevent self-oscillating (e.g. multiple short stimuli caused by clonus) and infinite-loop events (e.g. the subject can get locked in a transient position requiring stimulation). Such events

---

[13] An abstract and a version of this chapter were presented in the form of posters "Improved control for FES-aided locomotion after spinal cord injury", by Kostov et al., at the 10th Annual Spinal Cord Research Symposium organized by Canadian Paraplegic Association, Montreal, Oct. 27-28, 1994 and "FES-aided locomotion controlled in real-time by the artificial neural networks", by Kostov et al., at the 26th Annual Meeting of the Canadian Physiological Society, Mt. Tremblant, Jan 18 - 22, 1995. The abstract is published in *Physiology Canada*, Vol. 25, No. 2, p. 116 and accepted for publication in the *Canadian Journal of Physiology and Pharmacology*, March 1995. The paper containing most of this chapter "Automatic generation of FES-control rules: Application to real-time control of locomotion for subjects with incomplete spinal cord injury", by Kostov et al. (1995d), is in preparation for publication.

were expected to arise from the difference between walking controlled manually and walking controlled automatically (Stein et al., 1993). Although both adaptive logic networks and inductive learning have shown that they are able to learn very complex multidimensional mapping functions, they could not predict situations that result from new situations not seen during the training. We hypothesized that the generalization of manually controlled walking extends to automatically controlled walking as well.

3.6.1 Methods

Version of the ALN learning program: **ALN V.3** (see section 2.7.1 and Appendix B for details).

ALN interface program: **FESCONT** (see section 2.8.2 and Appendix C for details).

3.6.1.1 Subjects

Two subjects participated in this study, a person with incomplete spinal injury (L.W.) and a control subject who does not have any obvious walking abnormalities (A.K.). The development of the integrated control system required frequent presence of the control subject. The SCI subject who participated in this study (L.W.) is introduced in detail in the Section 2.1.2. Briefly, she has an incomplete quadriplegia due to an idiopathic hematoma at C1/C2 level. She has limited hand function, but can support her body weight when her knees are close to full extension. She can flex her right leg, but not her left leg while standing, because of spasticity. The stimulation is used to excite a flexion withdrawal reflex in her left leg assisting the swing of her left leg. She had a single-channel FES device implanted on the common peroneal nerve in the popliteal fossa area in April, 1990. In late 1993 she was implanted with percutaneous wires for multichannel FES positioned to stimulate: common peroneal nerve (induces ankle dorsiflexion and flexor withdrawal reflex); quadriceps muscle (knee extensor); psoas muscle (hip flexor); and gluteus maximus muscle (hip extensor). A single-channel percutaneous stimulation on common peroneal nerve controlled manually was used in the following experiment. The subject is able to walk more than 50 steps in

a single trial. She uses a four point wheeled walker for balance while standing or walking, and is skilled in both manual and automatic control of FES-assisted walking (see Section 3.2).

3.6.1.2 Experimental Design

The experimental setup used in this phase of the thesis project is illustrated in Figures 3.6.1 A and B. For the control subject the stimulator and electrodes were not used and the stimulation control was only simulated by the subject pressing on the manual switch whenever the left leg was about to start the swing phase during walking. For the SCI subject, the stimulator, electrodes and the manual switch are components that are in daily use for manual control of routine FES-assisted walking exercises. A single-channel percutaneous stimulation on the left common peroneal nerve was controlled manually in the following experiment as the basis for implementation of machine learning techniques in automatic generation of control rules for rule-based control of FES-assisted walking. Shoe insoles were used, each instrumented with three force sensing resistors (FSRs), positioned under medial and lateral metatarsal joints and under the heel. Instrumented insoles are described in details in Section 2.3.1. A signal conditioning device for force sensors is described in Section 2.4. The data acquisition system used in this experiment consists of an AT-MIO-16DH data acquisition board and LabVIEW for Windows software. Both components are described in detail in Section 2.5.2. The sampling rate used for ***ALN training*** and ***Off-line test*** was 20 Hz. It was reduced to 10 Hz during real-time ***Walking test*** and ***Walking control*** due to limitations of the program FESCONT. The program FESCONT, described in Section 2.8.2., was developed in the graphical programming environment LabVIEW for Windows. The program integrates all functions of the integrated control system required to implement the approach to designing a control system evaluated in previous sections. The program is still under development and currently it is in the form of a functional prototype. It uses a machine learning module implementing ALN V.3 for training on feedback sensory signals representing the ***domain*** and stimulation control signals representing the ***codomain***. The learning algorithm is presented in Section 2.7.1.

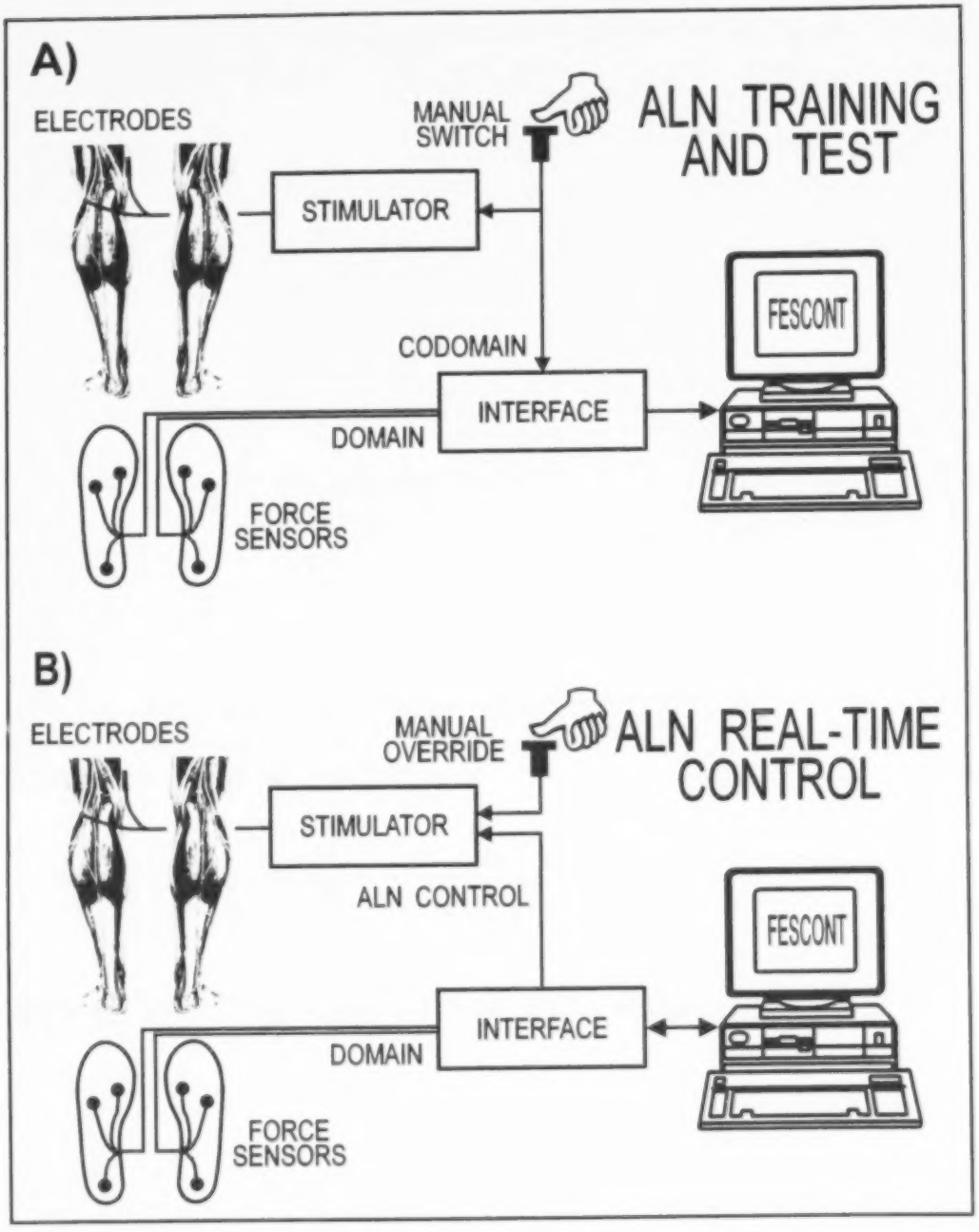

Figure 3.6.1. Integrated control system (ICS) for control of FES-assisted locomotion: A) **Data acquisition, ALN training** *and* **Off-line test**, *B) Generalization of trained ALNs in real-time* **Walking test** *and* **Walking control**.

The interface module contains: signal conditioning electronics for force sensors (see Section 2.4), a relay switch, which replaces the manual switch and which is controlled both from the manual switch and from the ALN control signal, a piezoelectric buzzer, and bidirectional connections to and from the data acquisition I/O system. Schematics of the signal interface and shoe insole sensors are presented in Figure 3.6.2. To provide feedback information to the subject and the researcher about the actions proposed by the controller, a small buzzer is installed in the interface box, which is driven from one of analog outputs of data acquisition I/O board.

Routine preparation for walking of the SCI subject consists of stretching the extremities, connecting all parts of the system together and checking the functionality of the ICS. The complete experimental protocol contains the following five steps related to the automatic generation of control rules and their application in real-time FES-assisted walking. For obvious reasons, the last step in the list, the real-time control of the FES-assisted walking could not be tested with the control subject.

1. ***Data acquisition:*** Input and output signals were recorded during several steps of manually controlled FES-assisted walking. If the walking represented a typical sample of the subject's walking pattern, i.e., if there were no problems with the fatigue or stimulation parameters, recorded signals were examined and we proceeded to the second step.

2. ***ALN Training:*** An ALN training was done usually on one half of the recorded data set. This part of the data is denoted 'training data set'. Trained ALNs are evaluated on the same data set and if the signal produced by ALNs was free of *functional errors* (missing pulses or extra pulses) the training was declared 'successful' and the ALNs were ready for the next step (see Figure 3.6.3A). If functional errors existed after evaluation on the training set (see Figure 3.6.3B), it was expected that there will be even more errors in the evaluation on new test data. In such a case, the training was repeated with a different training data set configuration and/or modified training parameters until the result of the evaluation on the training set was without functional errors.

3. ***Off-line test:*** Test of the trained ALNs was done on the second half of the recorded data set or on a data set recorded during another walking round. If the signal predicted by the ALNs, after being filtered through the restriction rules was free of functional errors, the ALNs were ready for the next step. If functional errors were present in the predicted signal, even after it was filtered by restriction rules, the previous step was repeated with a different configuration of the training data and/or modified training parameters until the predicted result was without functional errors.

4. ***Walking test:*** The walking test of the trained ALNs was done while the SCI subject controlled her stimulation manually as usual, but this time the ALN-based control system was delivering its predictions of the manual switching signal in the form of sound signals whenever the stimulation should be ON. The control subject simulated stimulation control as in the first step. This served to test the generalization of the trained ALNs on the same walking pattern once more and to demonstrate successful stimulation prediction to the subject. If this last test before the real-time application of ALN-control was successful (all stimulation trains correctly predicted), ALNs were ready for the next step. Otherwise, a sequence starting with step number two was repeated.

5. ***Walking control:*** In this step the ICS produces the actual stimulation which is responsible for the swing phase of the subject's walking. This step was completed only with a SCI subject. Real-time ALN-control was applied to the FES-assisted walking. The subject stands up from the wheelchair, takes one or more steps controlled manually to check the stimulation parameters (intensity and pulse duration) and if all components of the system operate satisfactory, the direct control of the stimulation by ALNs is turned ON. After that the subject's 'only' task is to initiate every new gait cycle by stepping first with her less impaired leg (right) and to shift body weight from left to right. That is enough for the controller to recognize the subject's intention to take another step with the paralyzed leg and the stimulation is delivered. The stimulation duration directly depends on the speed of weight transfer, the faster the weight transfer is - the shorter will be the interval between periods of stimulation. If the weight transfer to the right leg did not occur the stimulation will not be activated and the subject can rest in this position indefinitely.

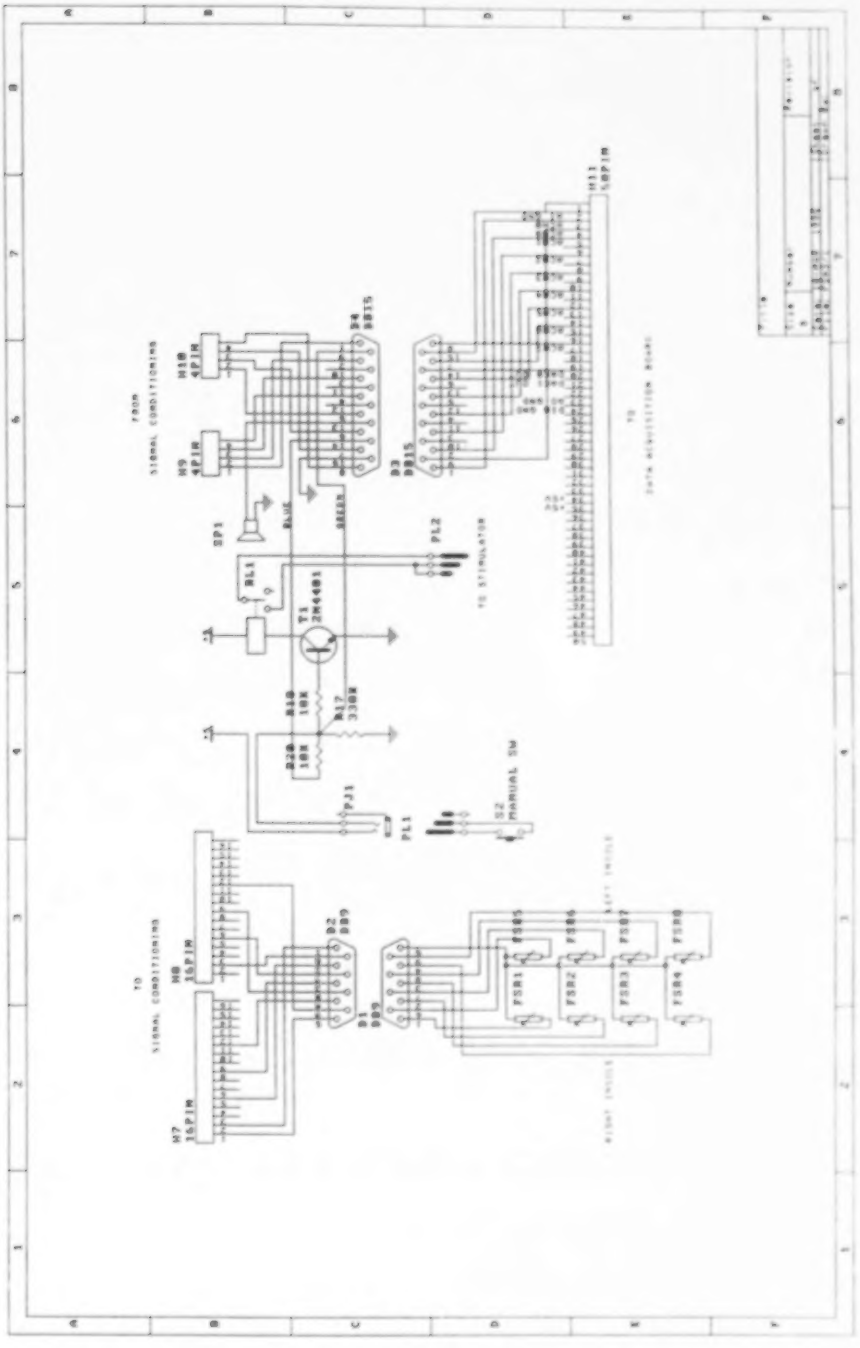

Figure 3.6.2. Electronic circuits interfacing stimulator, manual switch, sensors, signal conditioning devices, and PC-based ALN controller.

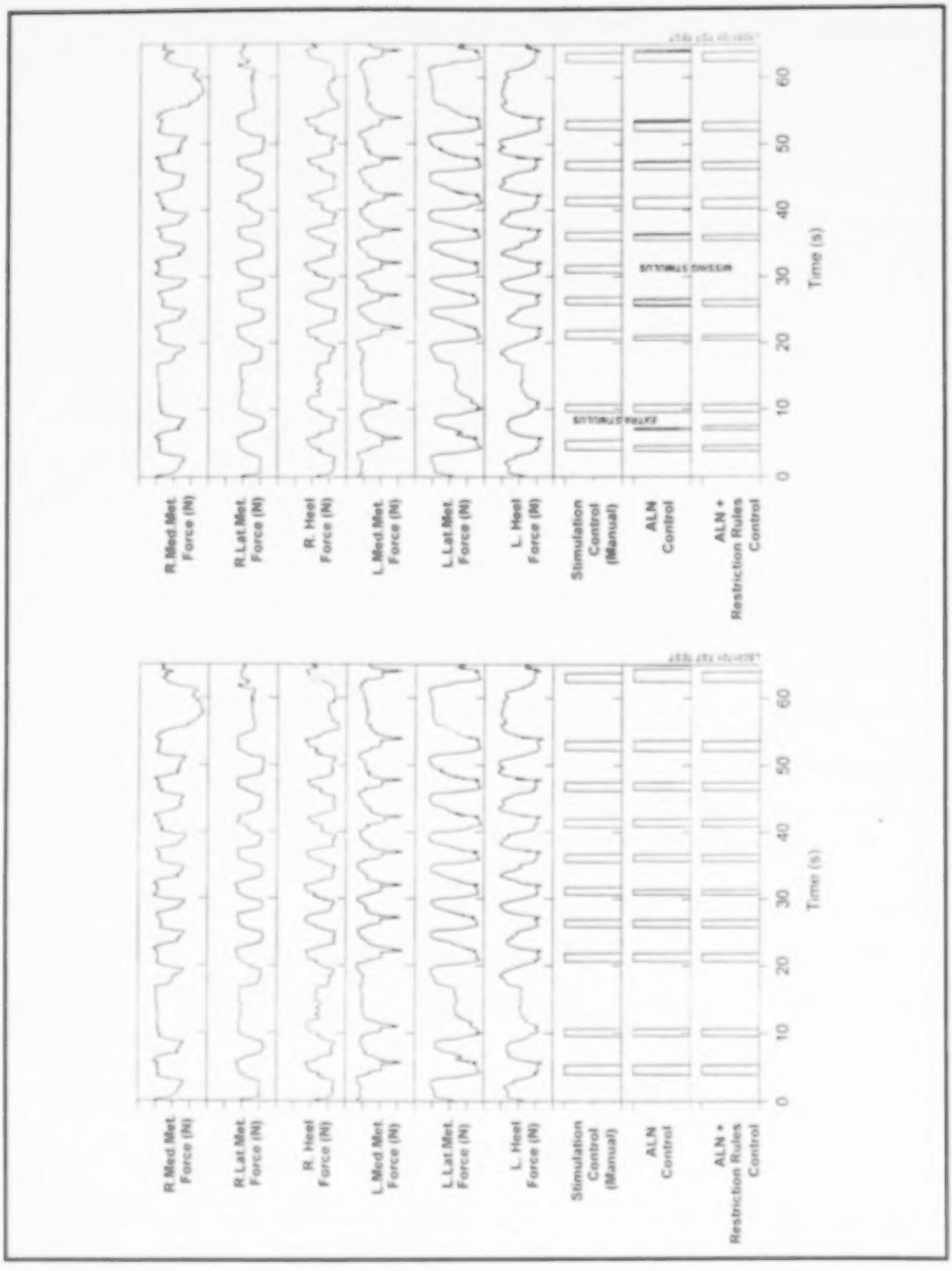

Figure 3.6.3. ALN training evaluation: A) ALN training evaluation with no functional errors; B) ALN training evaluation producing functional errors.

3.6.1.3 Safety concerns: Restriction rules

During FES-assisted walking the following situations were recognized as potentially dangerous:

- a new stimulus is delivered too soon after the previous one finished;
- a glitch was generated during the initial phase of the stimulation causing a short break in the stimulation train;
- due to improper weight shift from less impaired (right) to more impaired (left) leg, conditions to stimulate never end.

Such situations are usually caused by clonus, high muscle tone, fatigue, or weak physical conditions during recovery from acute health problems. Although being irregular, missing stimulation pulse train is not considered a high-risk situation because the subject can always deliver failed stimulation manually. Broken, extra or too long stimulation pulse trains are more dangerous because they may throw the subject out of balance and cause falls.

To provide additional safety for the subject, the following *restriction rules*, based on statistical measurements in the training data set were implemented:

1. The interval between two stimuli can not be shorter than the shortest interstimulus interval from the training set reduced by two standard deviations.

2. The stimulation pulse train can not be shorter than the average pulse train from the training set reduced by two standard deviations.

3. The stimulation pulse train can not be longer than the average pulse train from the training set increased by two standard deviations.

Although these rules provided required additional safety for the subject, they are quite arbitrary, which leaves plenty of room for their optimization. Optimization of restriction rules was not an objective of this project.

3.6.2 Results and discussion

A standardized pattern for data presentation is designed for the results in the following text of this Section. ***Nine-trace*** diagrams are used to present results after completion of ***ALN training*** (step two) and ***Off-line test*** (step three). The upper six traces in the diagrams are feedback force sensory signals. Other three traces are: simulated (subject A.K.) or real (subject L.W.) manual control of the stimulation (seventh trace), and the results of the evaluation of trained ALNs on the feedback sensory signals in the same data set, before (eight trace) and after (ninth trace) restriction rules are applied, respectively.

The results after completion of the ***Walking test*** (step four) and ***Walking control*** (step five) are presented using ***eight-trace*** diagrams. The first six traces are the same as in the nine-traces diagrams. The seventh trace is the actual control signal that drives the stimulator, produced either manually or by the ALNs. To distinguish between pulses produced manually and those produced by ALNs, manual pulses have a higher amplitude than those produced by ALNs. The amplitude of the pulses does not have any significance for the stimulator, i.e. both amplitudes produce the same stimulation intensity. The last trace is the ALN prediction of the stimulation control signal, after being filtered through the restriction rules.

The results of this experimental work are described qualitatively rather than quantitatively, because currently there are not enough subjects and data to perform any statistical analysis. In the future the ICS will be applied to more subjects and a quantitative analysis of the results will be performed. The criterion used for deciding on the continuation of the experiment or repeating the previous step was the presence of several functional errors, i.e. missing and extra stimuli.

3.6.2.1 Control subject (A.K.)

The following three figures illustrate ALN Training, Off-line Test and Walking Test on signals recorded during walking by subject simulating manual control of the stimulation.

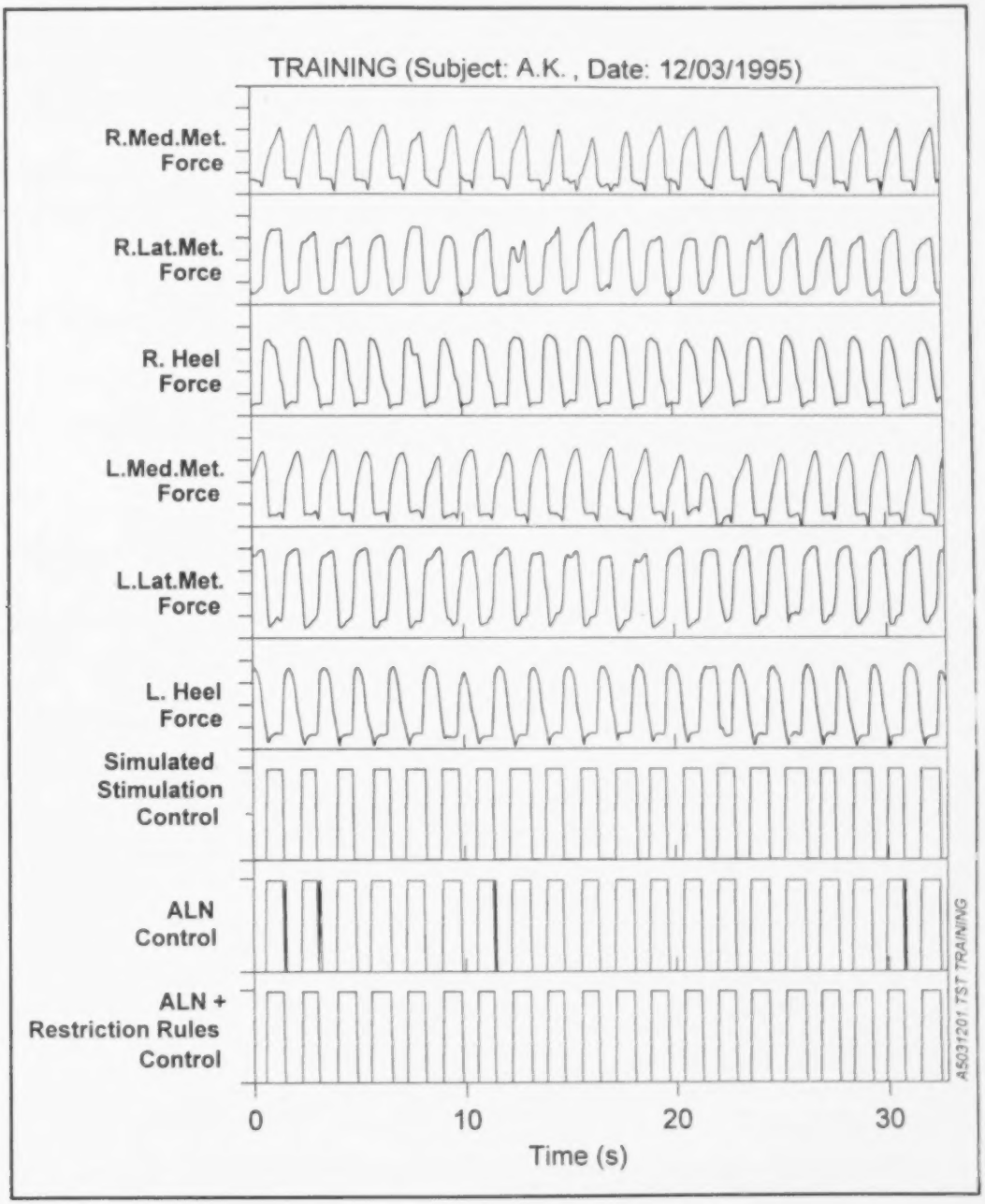

Figure 3.6.4. ***ALN training*** *results applied on the training data set recorded in a control subject (A.K.). The subject was asked to simulate the stimulation signal required to activate flexion in his left leg. The simulated stimulation control signal (seventh trace) was produced by pressing on a manual switch. The two bottom traces are results of the evaluation of trained ALNs before and after restriction rules were applied. There are no functional errors (missing or extra stimuli) in the obtained output control signal (bottom trace).*

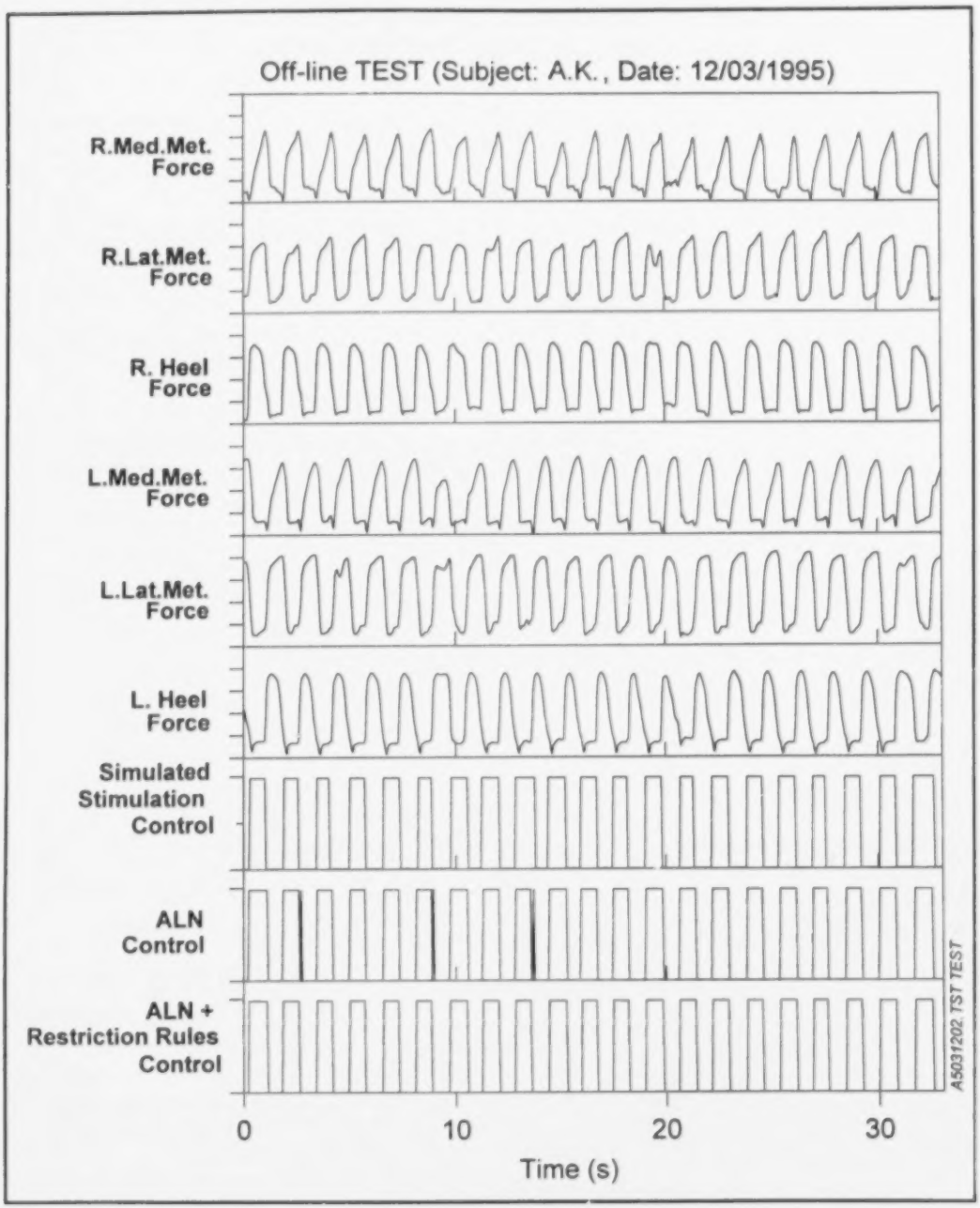

Figure 3.6.5. **Off-line test** *results on a new sequence of data recorded in a control subject (A.K.). There are no functional errors in the obtained output control signal (bottom trace).*

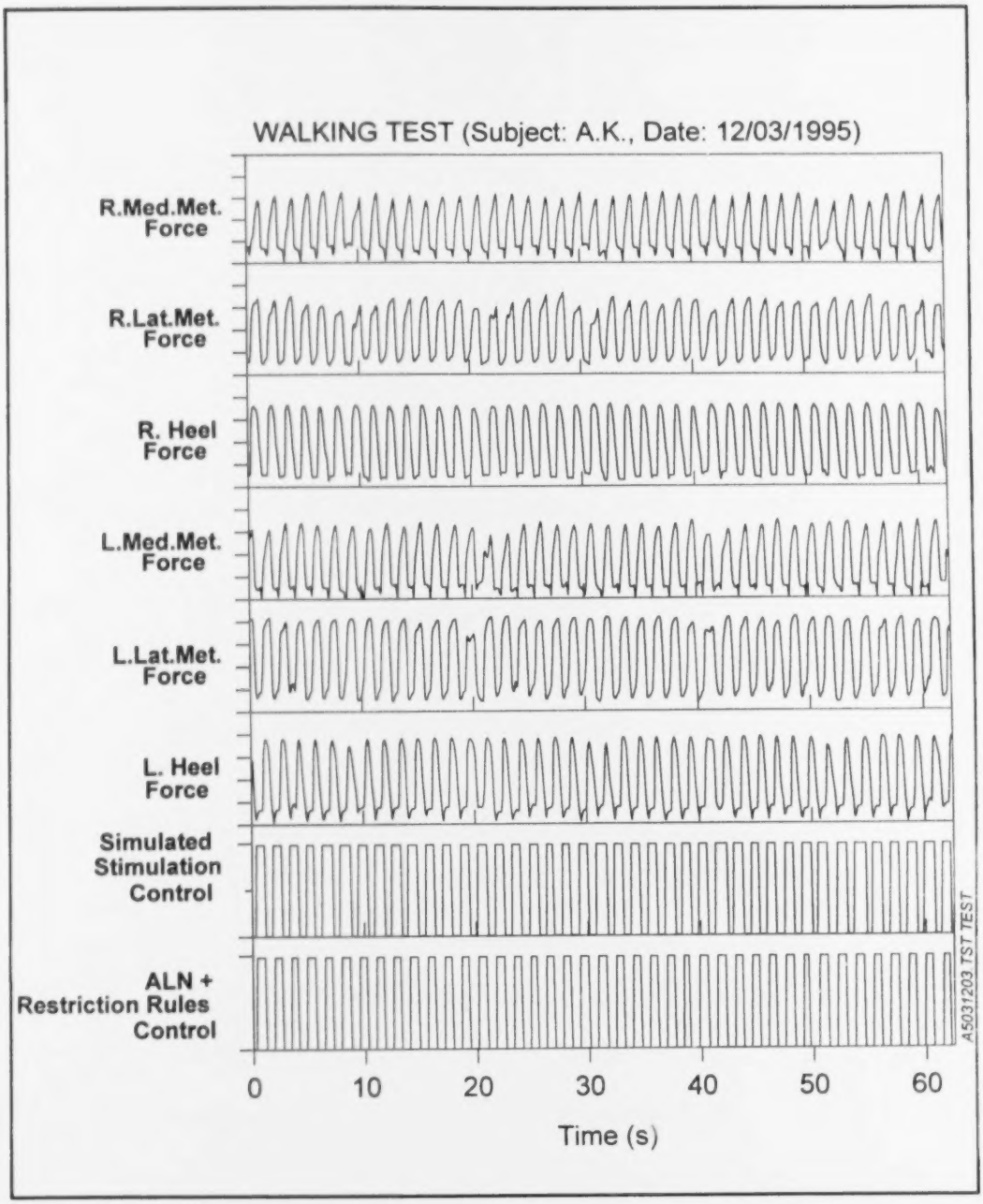

Figure 3.6.6. **Walking test** *results obtained by real-time evaluation of the trained ALNs during walking. A stimulation control signal predicted by the ALNs was presented to the subject in the form of a tone signal which sounded whenever the stimulator should be ON. As in previous steps, the subject was asked to simulate the stimulation signal required to activate flexion in his left leg.*

3.6.2.2 Subject with incomplete SCI (L.W.)

Most of the experimental work related to the development of the ALN-based Integrated Control System (ICS) was done between May 1994 and the end of October 1994. Very soon after the system was integrated and first satisfactory ALN training and generalization over the same session were obtained, we stopped the experimental work temporarily due to an acute illness of the subject. Walking experiments with the ICS were continued in early March 1995, and the subject needed a few sessions to bring her skill back to where it was before the illness. In the following text only the results obtained after the four month pause are presented, demonstrating restoration of the subject's motor skills, from the first insecure walking steps controlled manually to a smooth walking, fully controlled by ALN-based ICS. Three walking sessions are presented.

Session one

The data acquired during the first session after the four-month pause are presented in Figures 3.6.7 and 3.6.8. The subject stood up from the wheelchair without stimulation and took a dozen steps (about 7 m) manually controlling stimulation. Then she stopped and made a turn around her disabled leg and walked back the same distance. The subject turned back by pivoting around her disabled leg. She is also trained to use the stimulation to move her disabled leg backwards, which usually shortens the time required for turning around and reduces fatigue. In this experiment the first half of the data recorded is used for ***ALN training*** (Figure 3.6.7) and the second half is used for ***Off-line test*** (Figure 3.6.8). Duration of the ALN training was only 35 s. In the ALN Control trace of Figure 3.6.7 a double or 'twin' stimuli produced by the subject can be seen in the second step. Since this was just a single case, it did not influence any of the periodic steps in the central part of the diagram. It probably helped to produce the second stimulus in the last step. However, both of these twin stimuli were successfully removed by restriction rules and the output control signal resembles almost perfectly the manual stimulation control. The development of fatigue can be seen in the Figure 3.6.8 where the walking is not as continuous as in the Figure 3.6.7.

After the successful ***ALN training*** and ***Off-line test***, the walking test of the trained ALNs was performed. The subject walked over the same distance (7 m), controlling the stimulation manually (see Figure 3.6.9). The computer program FESCONT acquired sensory signals, and applied the trained ALNs and restriction rules on them, producing an output control signal. The output control signal also drove a small piezoelectric buzzer which produced a tone every time the ALNs predicted that the stimulation should be ON.

After a successful ***Walking test***, a real-time evaluation of the ALN-based control was performed. The subject stands up from the wheelchair, takes one or more steps controlling the stimulation manually (as during the ***Walking test***), and then the output control signal predicted by ALNs and filtered by the restriction rules is connected to the stimulator as the primary control system, in parallel to the manual switch. The switch enables the subject to keep her manual override function in case of missing stimuli. In the current system the problem of extra stimuli produced by ALNs remains unresolved. ***Walking control*** tests are presented in Figures 3.6.10 and 3.6.11.

Session two

As with the first walking session, signals are recorded during manually controlled walking which includes walking in one direction, turning by pivoting around the more disabled leg (no stimulation) and walking back. The first part (walking straight) was used for ALN training (Figure 3.6.12), while the Off-line test was done on part including turning and walking back (Figure 3.6.13).

After the successful ***ALN training*** and ***Off-line test***, the ***Walking test*** of the trained ALNs was performed (Figure 3.6.14). After the successful ***Walking test***, a real-time evaluation of the ALN-based control was performed (Figure 3.6.15). After standing up from the wheelchair, the subject took two steps controlling the stimulation manually (same as during the walking test), and then the output control signal predicted by ALNs and filtered by the restriction rules was connected to the stimulator as the primary control system, in parallel with the manual switch.

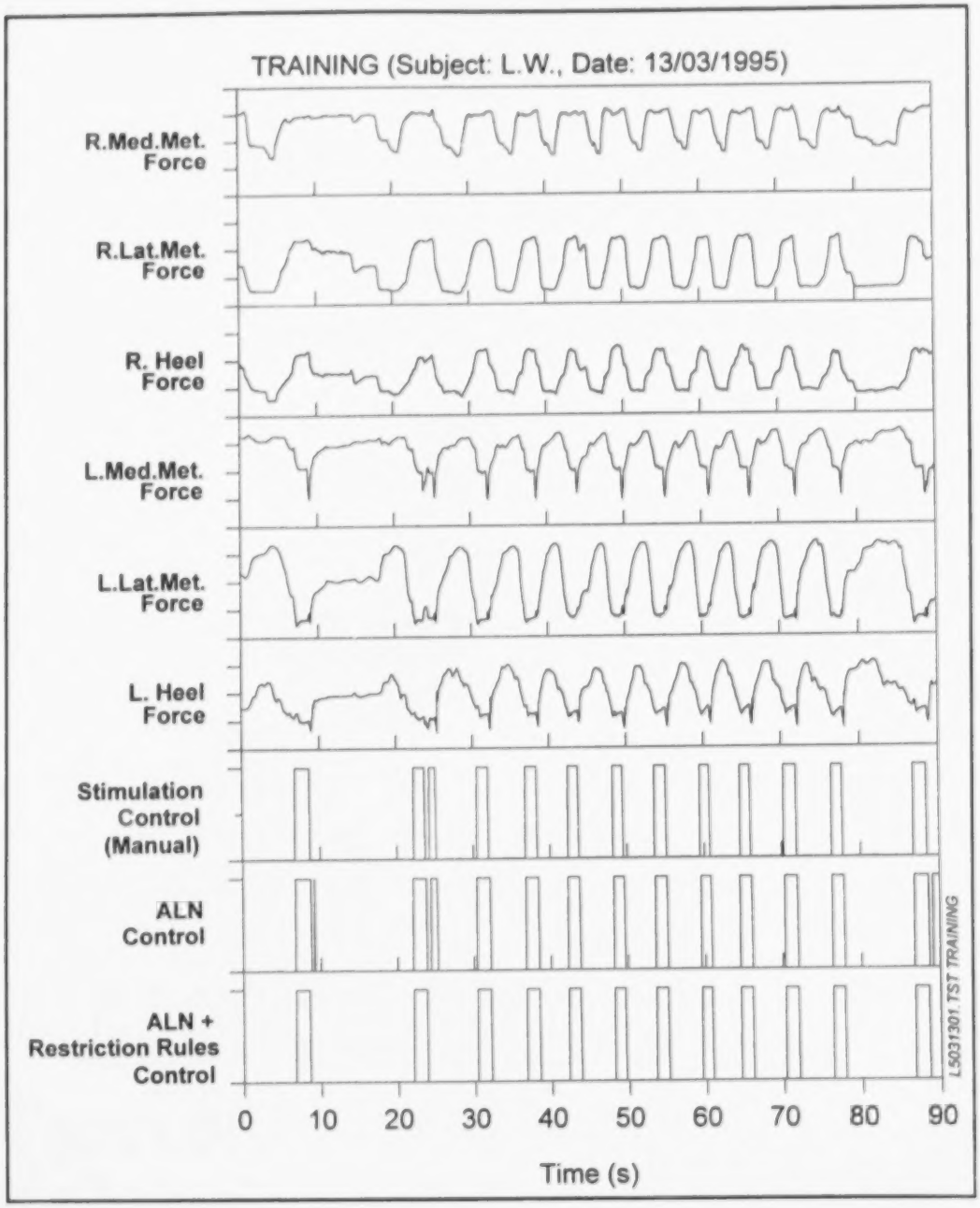

Figure 3.6.7. **ALN training** *results applied on the training data set recorded in an SCI subject (L.W.). The subject controlled the stimulation by pressing on a manual switch installed on a grip of her four-point wheeled walker. The stimulation control signal (seventh trace) was used to activate a flexion withdrawal reflex in her left leg. The two bottom traces are results of the evaluation of trained ALNs before and after restriction rules were applied.*

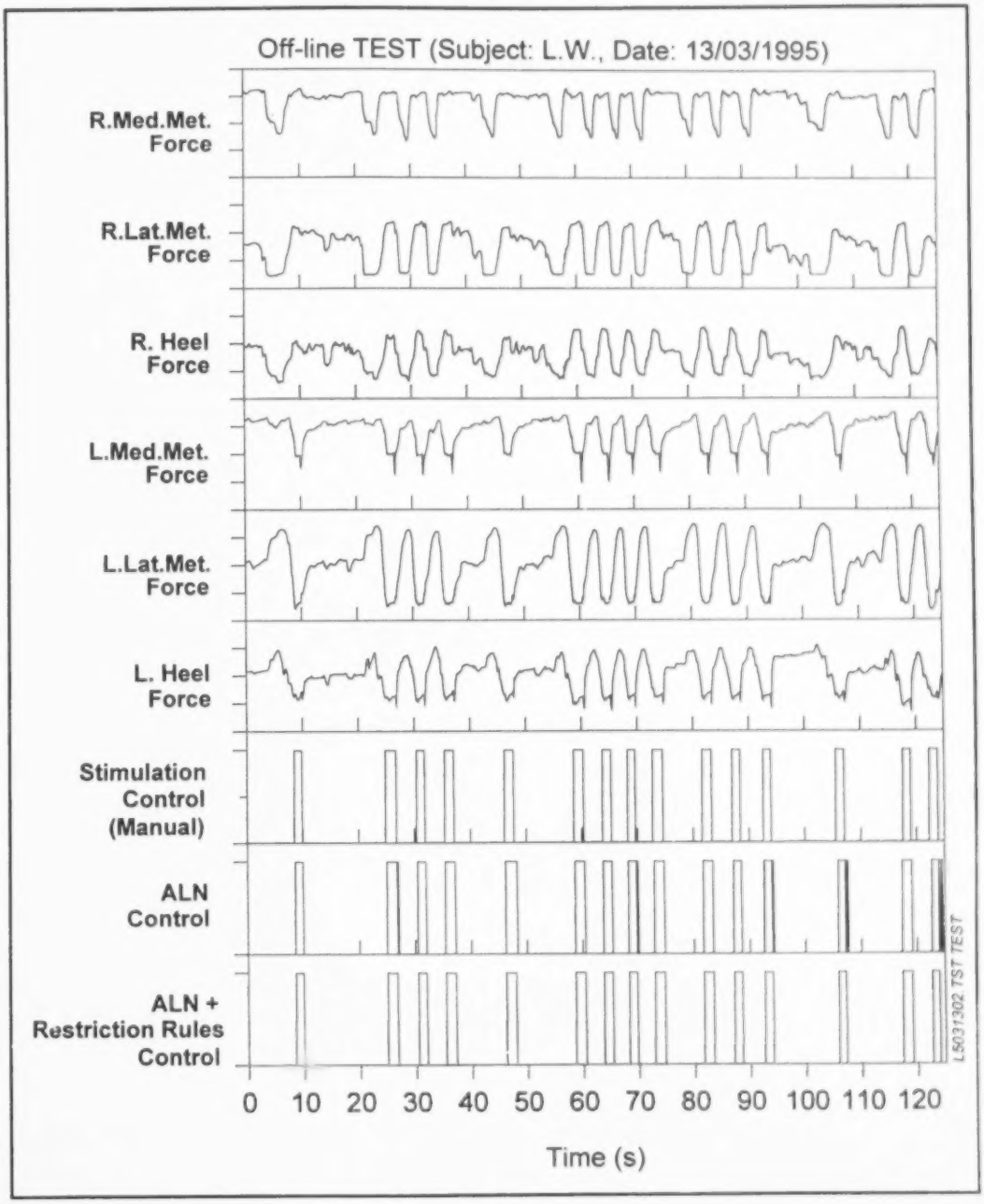

Figure 3.6.8. **Off-line test** *results on a new sequence of data recorded in an SCI subject (L.W.). Although there are some glitches at the end of several stimuli in the second trace from the bottom, there are no functional errors in the obtained output control signal (bottom trace) due to application of restriction rules.*

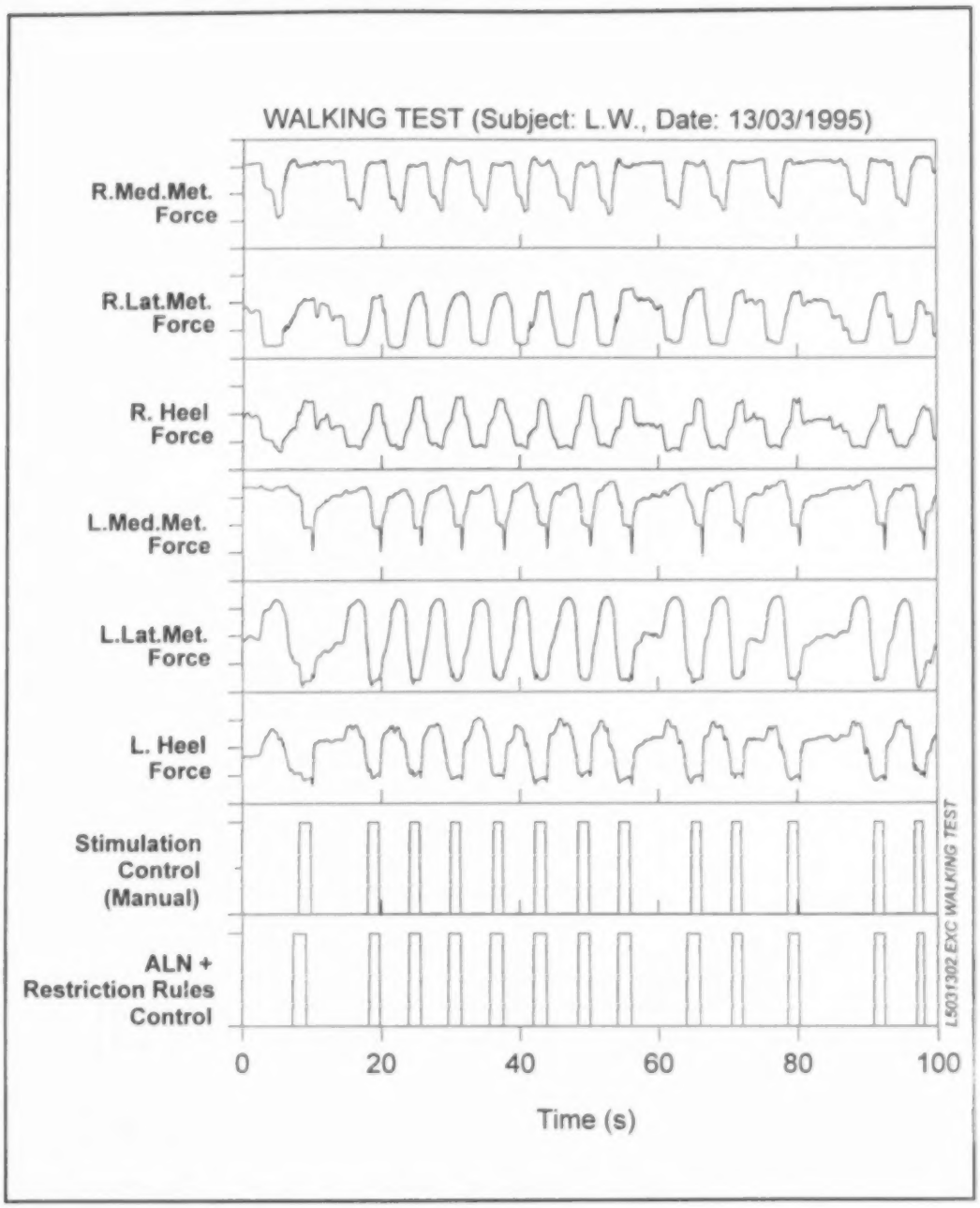

Figure 3.6.9. **Walking test** *results obtained by real-time evaluation of the trained ALNs during walking. The stimulation control signal predicted by the ALNs was presented to the subject in the form of a tone signal which sounded whenever the stimulator was supposed to be ON. As in previous steps, the subject was asked to manually produce the stimulation signal required to activate flexion in her left leg.*

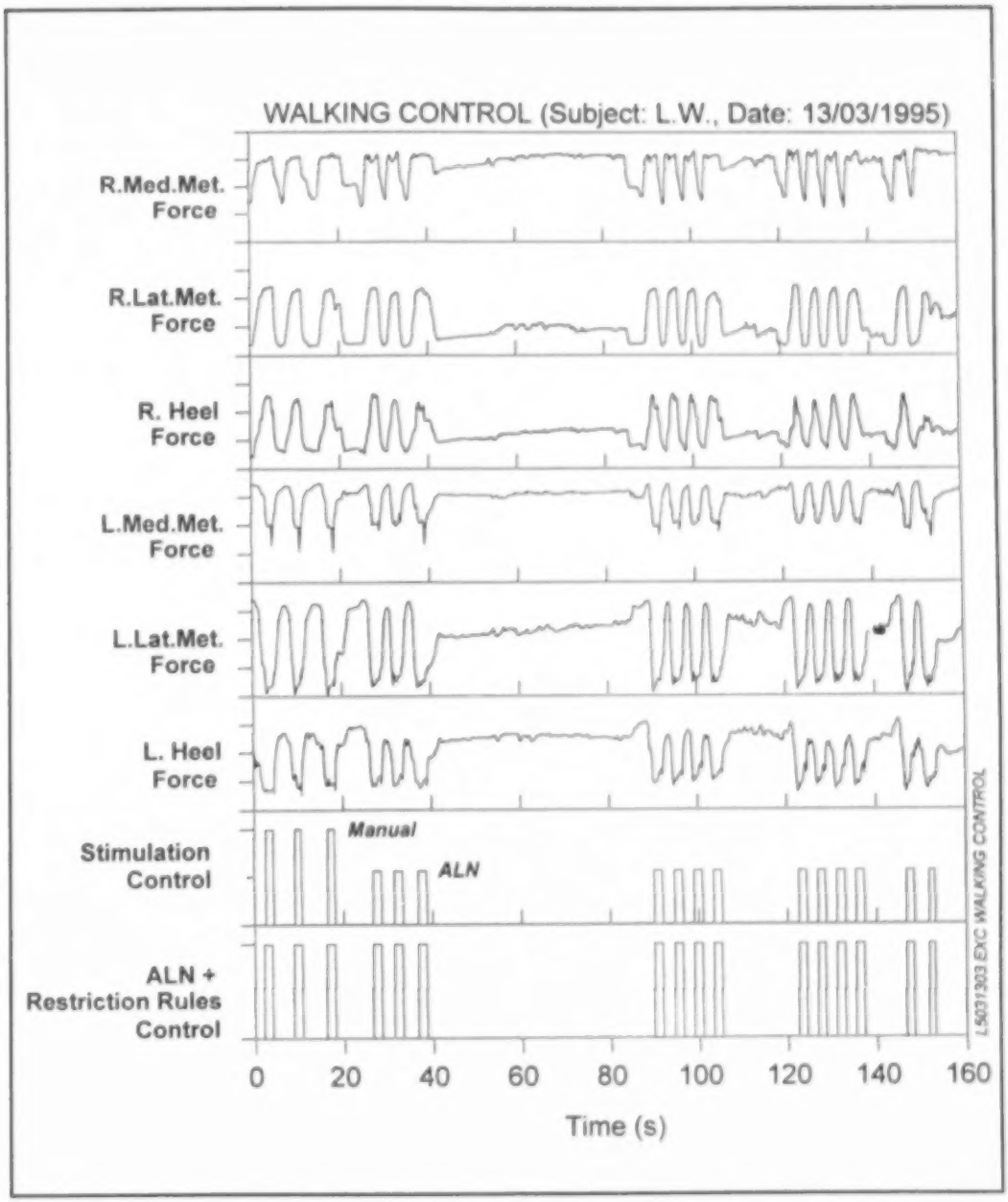

Figure 3.6.10. **Walking control** *test performed after three steps that the subject controlled manually. Not being physically fit for walking, and having not walked because of an acute illness for several months, the subject was very tense which was reflected in the quality and number of steps taken. This was the reason behind having short walking sequences with long resting periods. After the walking session the subject admitted that she carried more than the usual part of her body weight on her upper extremities, which resulted in continuous turning of her walker to the right. That was another reason to stop after every few steps to correct the direction of walking and the walker. There were no functional errors in this walking session.*

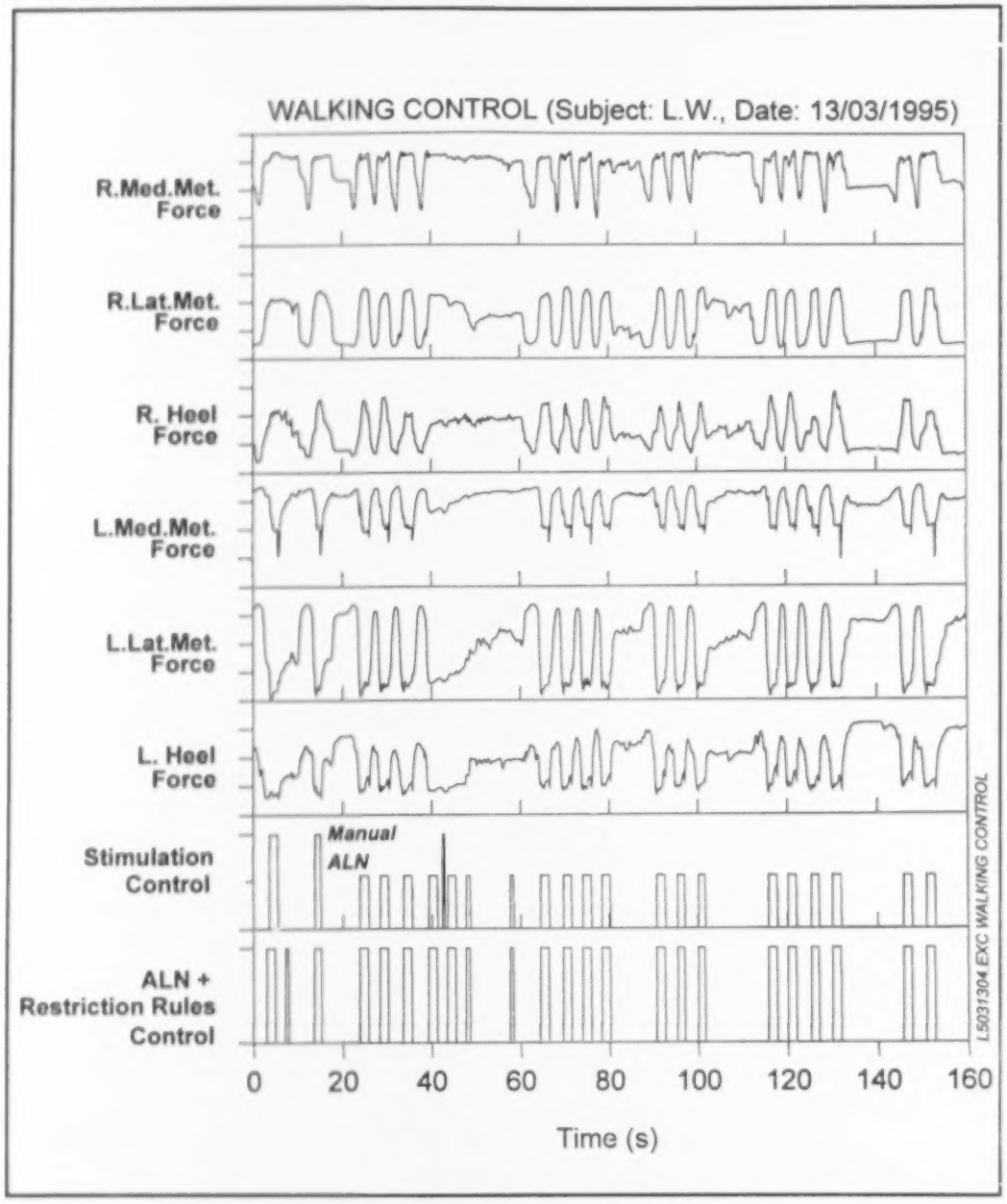

Figure 3.6.11. The second **Walking control** *test demonstrates better subject's confidence in the control system, longer walking sequences and shorter resting intervals. Also, there was less load on the subject's arms, which resulted in less turning of the walker and a straighter walking path. After the sixth step (two manually and four ALN controlled), the subject accidentally, pressed on the manual switch. This caused instability and two extra stimuli (see arrow). Although a situation such as this one is potentially dangerous, we helped the subject to restore her balance and she continued walking. An additional restriction rule remains to be added which would prevent events such as this, but at the time of the experiment it did not exist.*

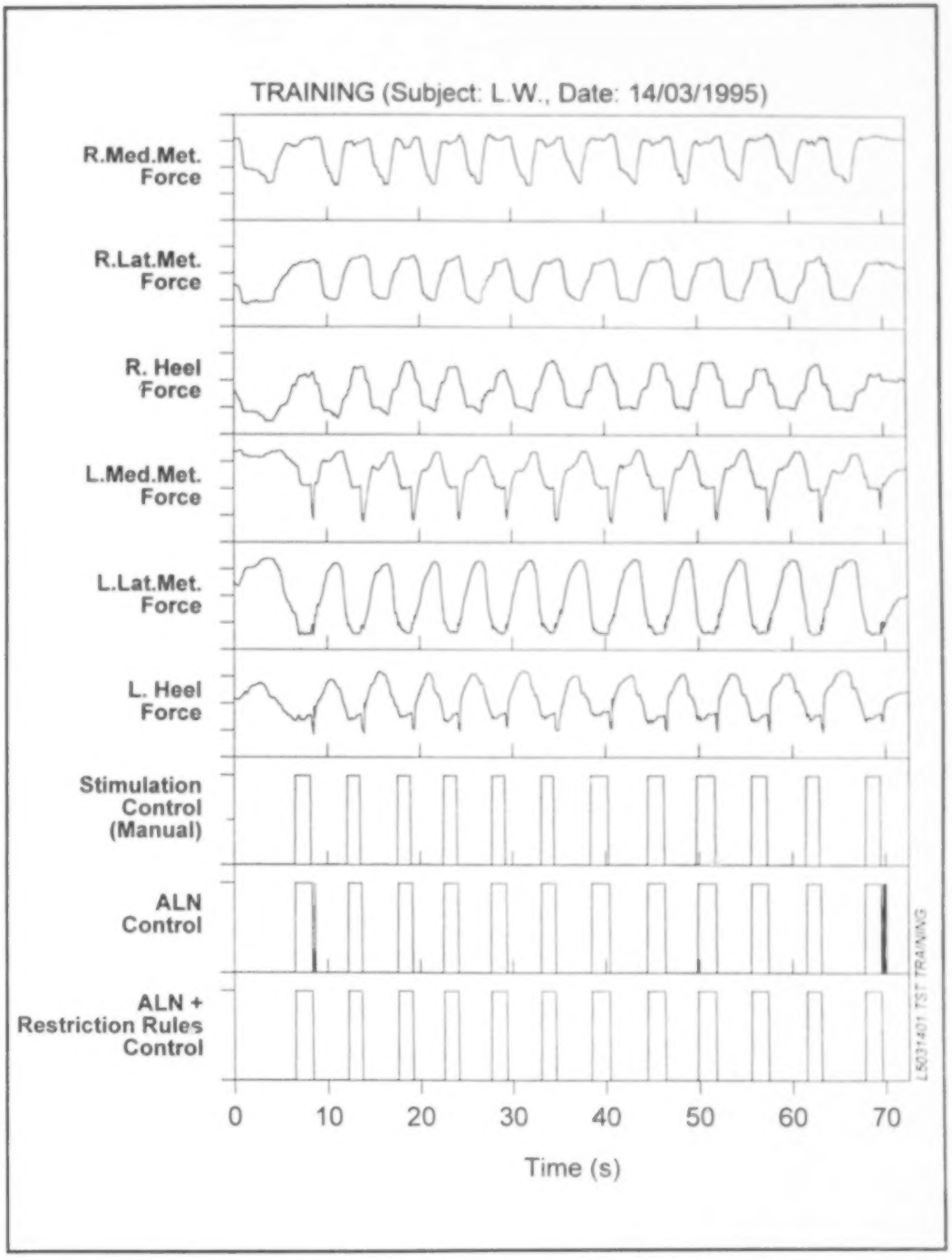

Figure 3.6.12. **ALN training** *results on signals recorded during manually controlled walking in one direction. No functional errors were generated by ALNs and all stimuli were approximated successfully.*

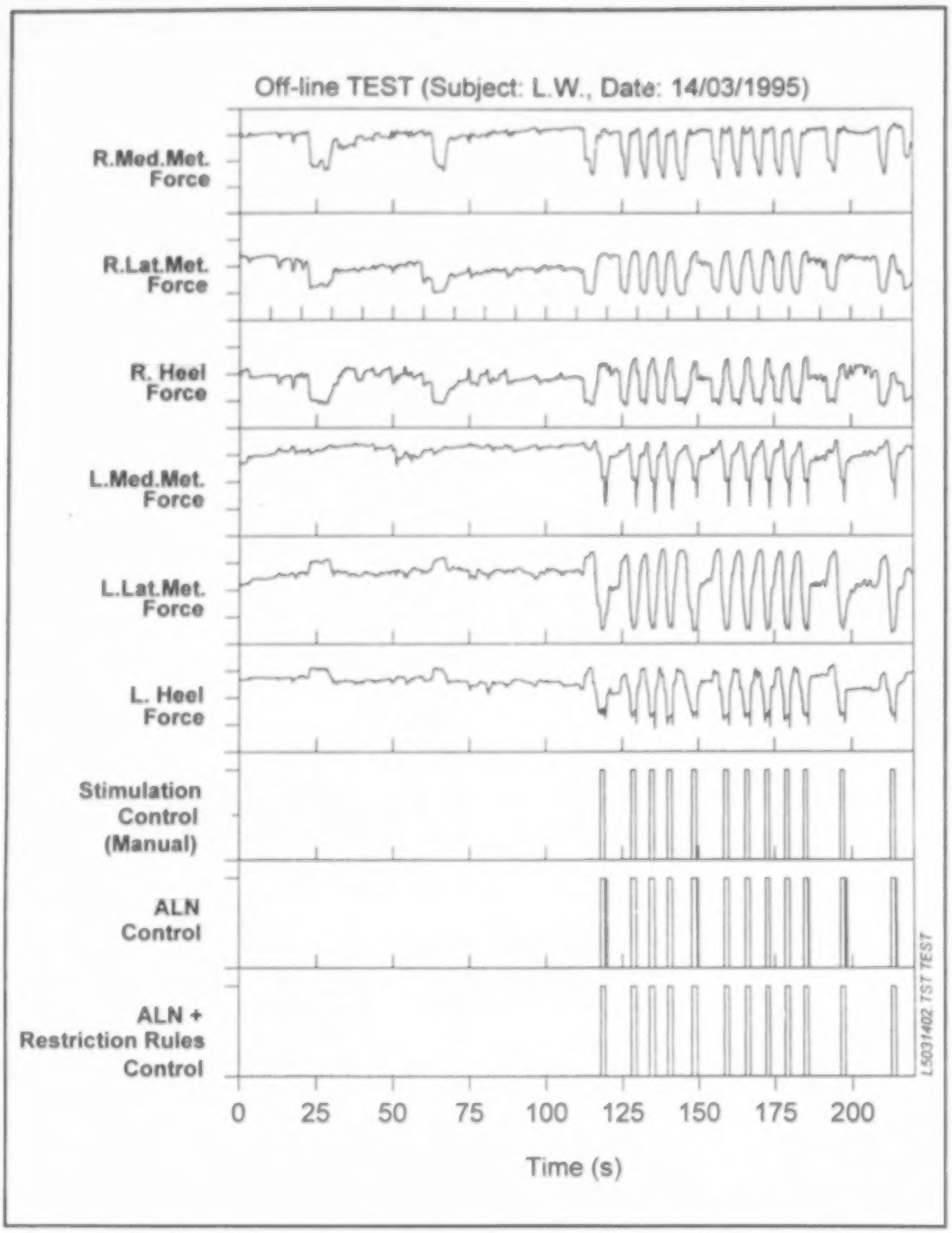

Figure 3.6.13. **Off-line test** *of the trained ALNs on the second half of the same walking round used for training. Although turning around was not part of the training data set, ALNs generalized well during turning around without stimulation and during walking back to start position. No functional errors or obvious dissimilarities between the original stimuli and ALN-produced ones are present.*

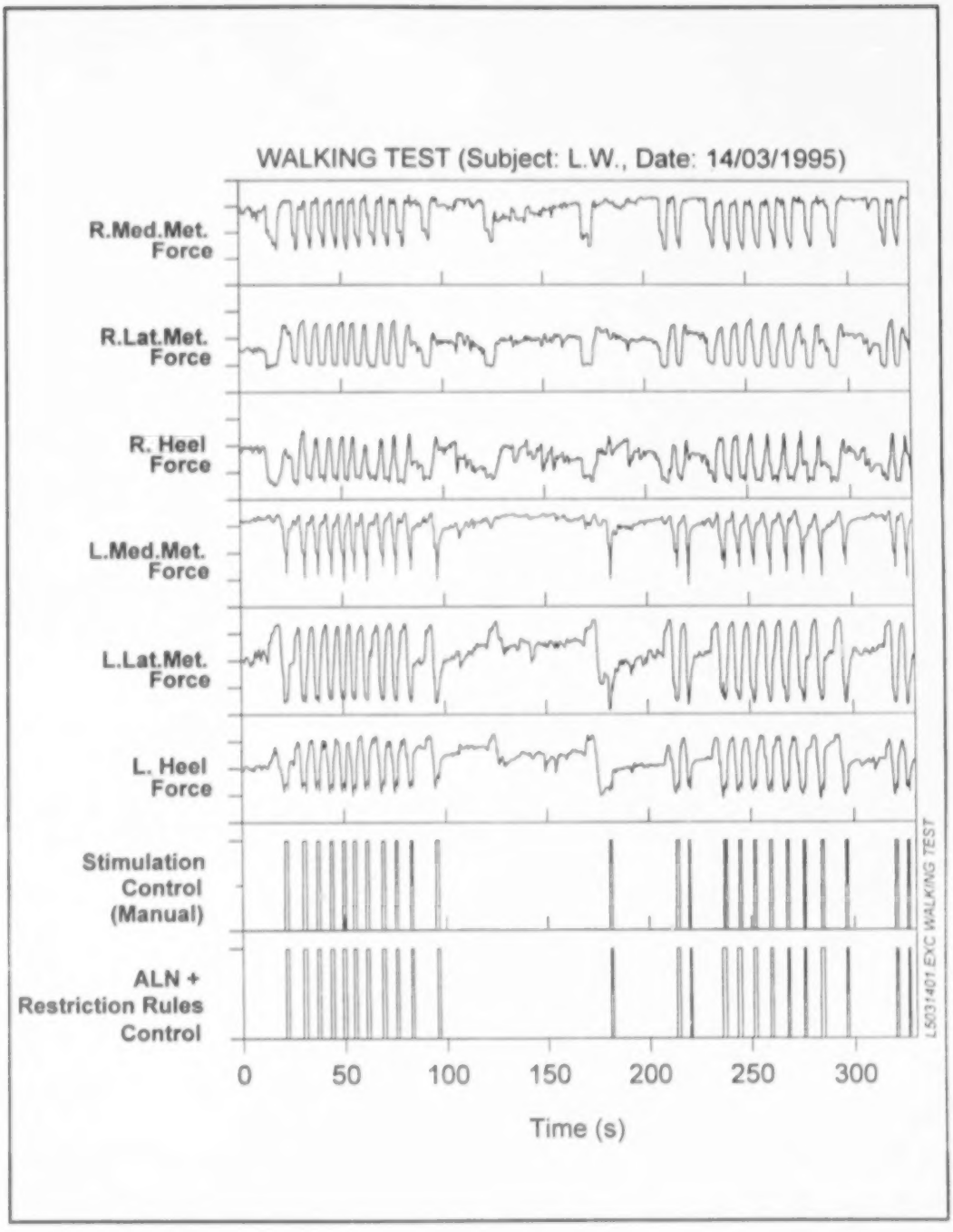

Figure 3.6.14. **Walking test** *results obtained by real-time evaluation of the trained ALNs during walking. The subject used stimulation not only for walking straight, but also during turning around for taking steps backwards. ALNs generalized well and without functional errors both during straight walking and during turning around.*

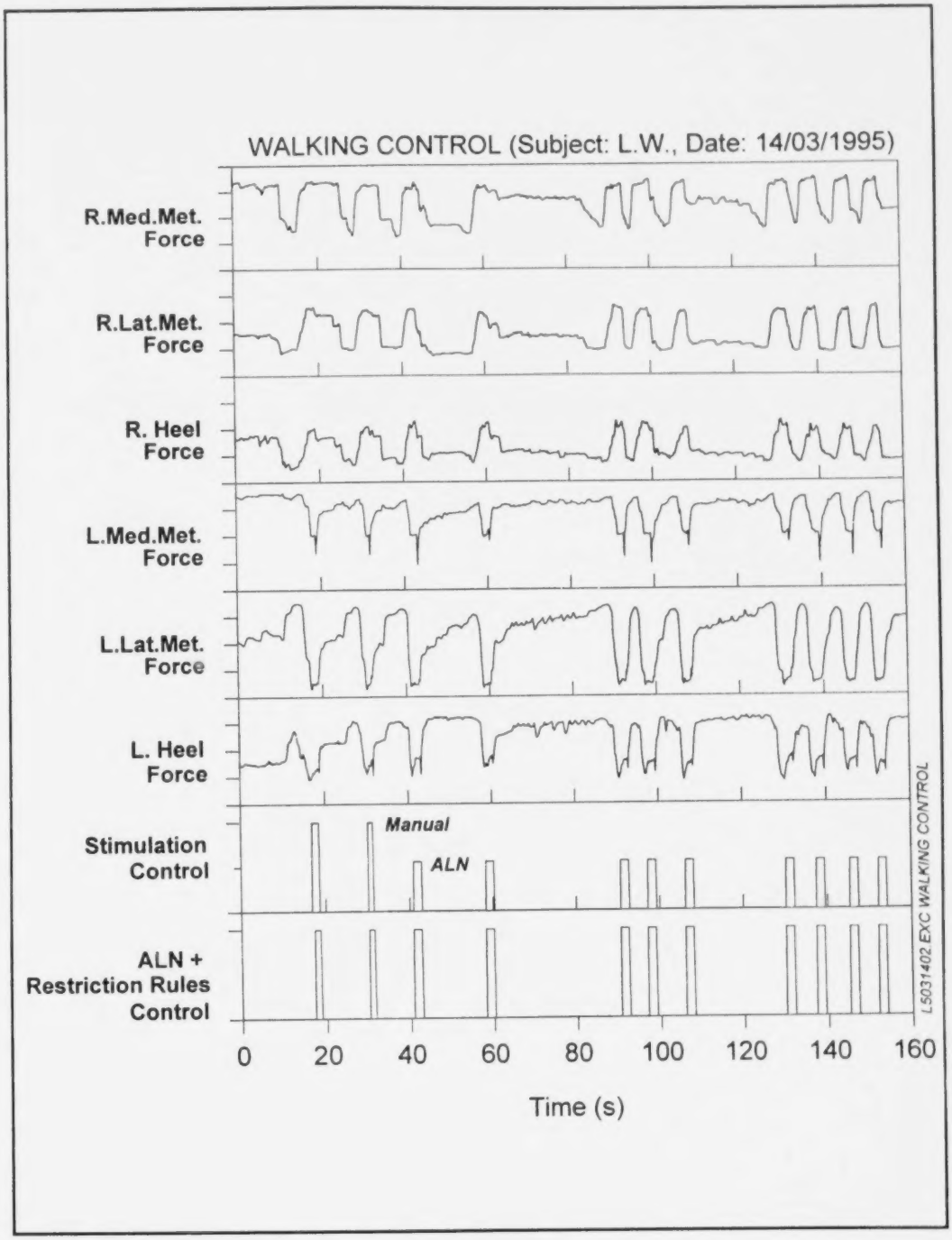

Figure 3.6.15. **Walking control** *test performed after two steps controlled manually by the subject. This walking sequence was also very short due to the subject's fatigue. There were no functional errors in this walking session.*

Session three

Previous two walking sessions, as well as the previous sections presenting results from the evaluation of the machine learning techniques, demonstrated a good generalization of the trained ALNs on the new data set recorded during the same walking session. These results also positively answered the question whether the rules generated from data recorded during manually controlled walking can generalize to the automatically controlled walking. The next question answered in the remaining text of this section is if the rules generated from data recorded in one session can generalize to the walking done after large time periods (days, weeks, months). In practical terms, the question is: How often do the ALNs have to be trained or retrained depending on the time expired after the last training or changes and modifications in the system. In previous sessions we already concluded that taking the shoes and sensory insoles off and putting them on usually did not change the setting significantly and did not require retraining. This situation usually occurred in the middle of the experiment, or between walking rounds, when the subject had to use the bathroom.

The same ALNs trained and tested in session two were used three days later. This time only last two steps of the experiment were done: ***Walking test*** (step four) and ***Walking control*** (step five). The first ***Walking test*** (Figure 3.6.16) still looks insecure and in the first step it even has an extra stimulus, but it stabilizes nicely and in the second walking test (Figure 3.6.17) shows subject's full confidence in the system and very nice periodic walking. It is obvious that the ALN-predicted stimuli are shorter than the original ones. However it is too early to conclude that this may result in a problem for real-time walking control, because of the differences between manually and automatically controlled walking.

After the ***Walking test***, a real-time ***Walking control*** was done in the next three walking rounds (Figures 3.6.18-20). Same as described in previous two walking sessions, the subject stands up from the wheelchair, takes two or more steps controlled manually and then the ALN-control is

turned ON. Although there are some signs of insecurity at the beginning of all walking control rounds, after the ALN control is turned ON and the subject regards it as the primary control system of her stimulation, the walking becomes periodic, and the steps become very reproducible.

The three ***Walking control*** rounds clearly demonstrated that the ALNs do not need to be retrained at the beginning of every walking session. In addition, we have learned that due to differences between manually and automatically controlled walking, even if a walking test of trained ALNs does not look perfect, the walking control can be very good. This is probably due to the flexibility of the subject's preserved control mechanisms.

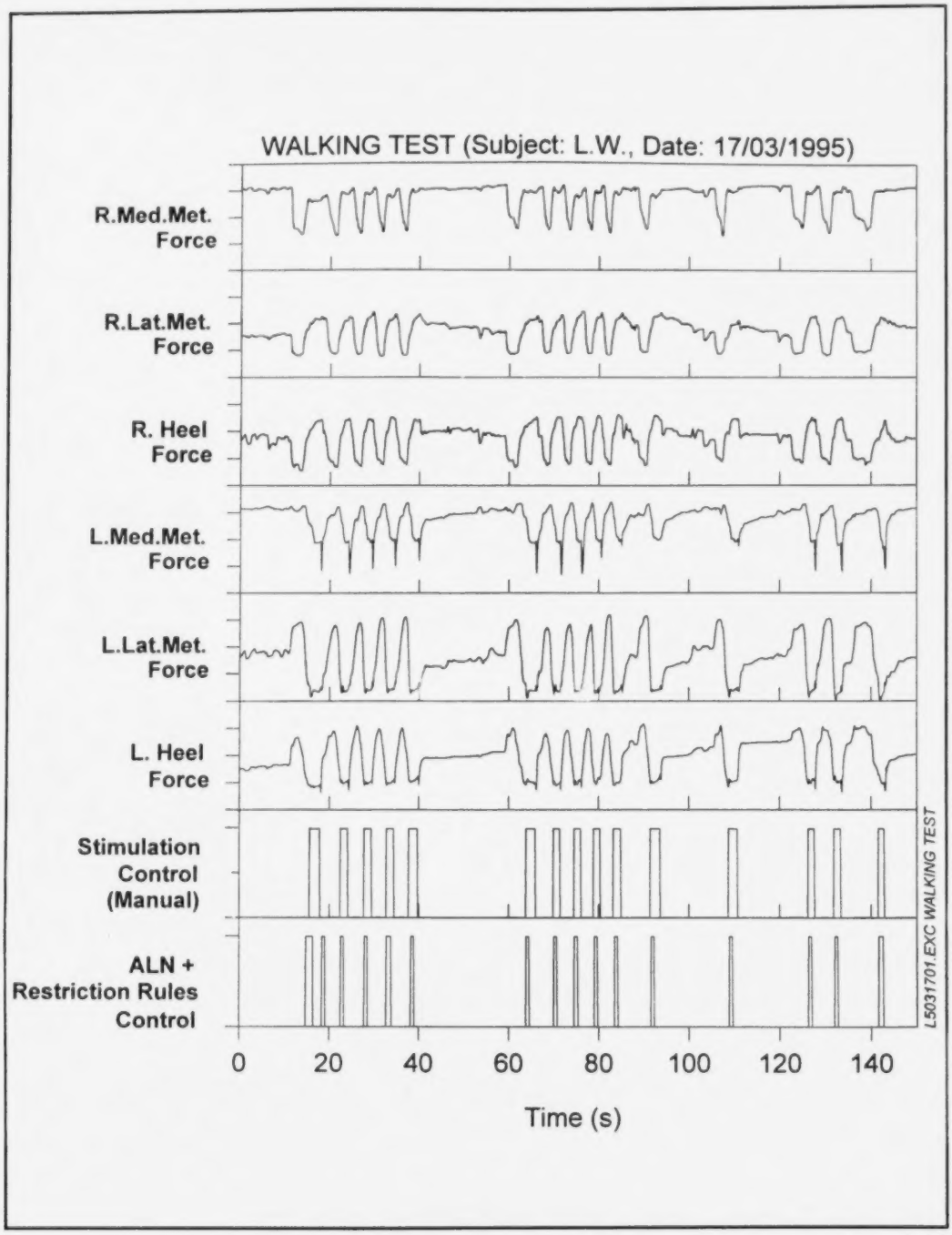

Figure 3.6.16. **Walking test** *done using ALNs trained during the Second session (tree days before this experiment). This was the first walking test of the ALNs trained before the current walking session. After initial insecurity (one early and one extra stimulus during the first step), the rest of the stimulation was predicted correctly.*

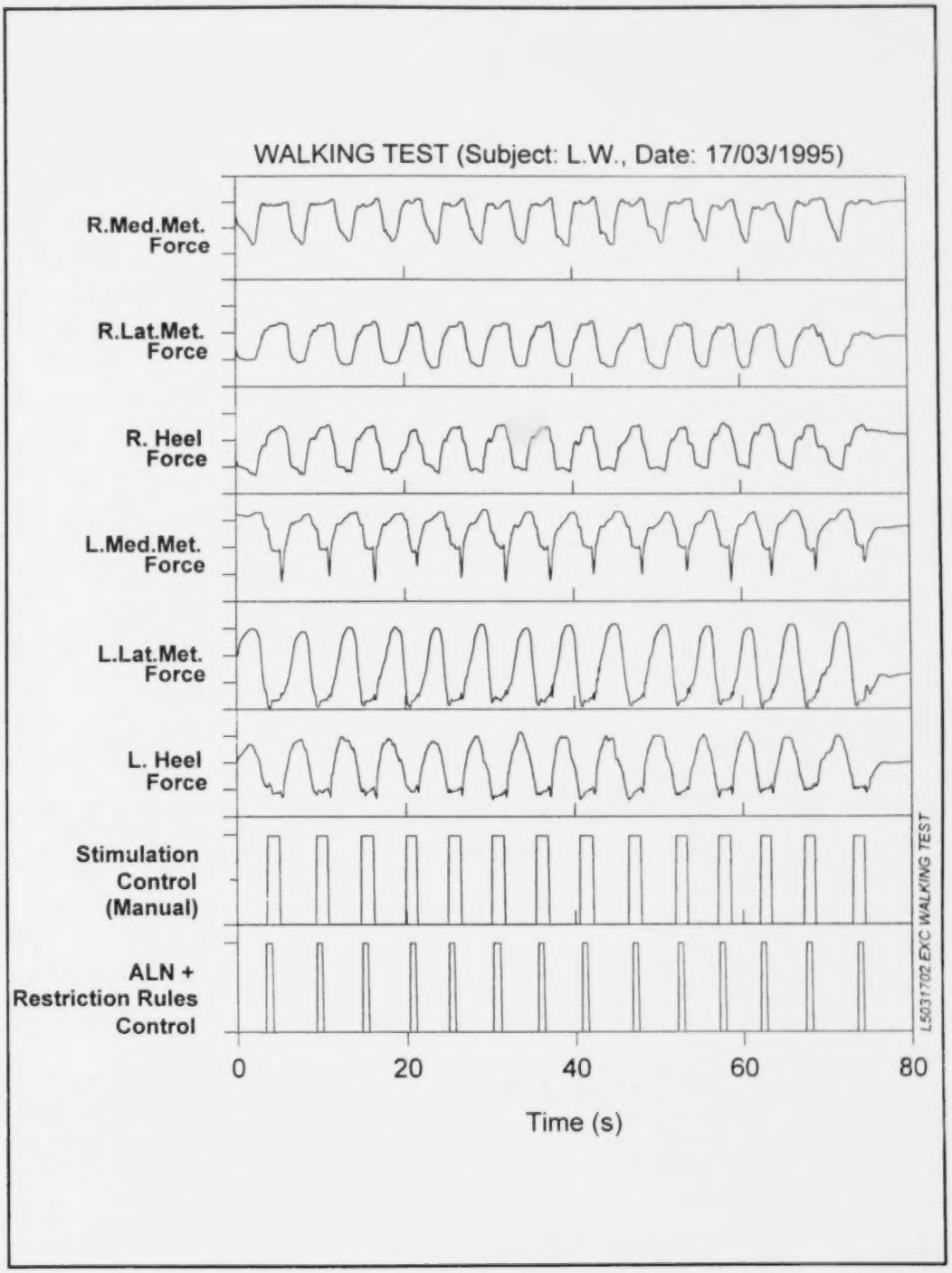

Figure 3.6.17. The second **Walking test** *done right after the first one. Obviously, the subject demonstrated nice periodic walking during which all stimuli were correctly predicted.*

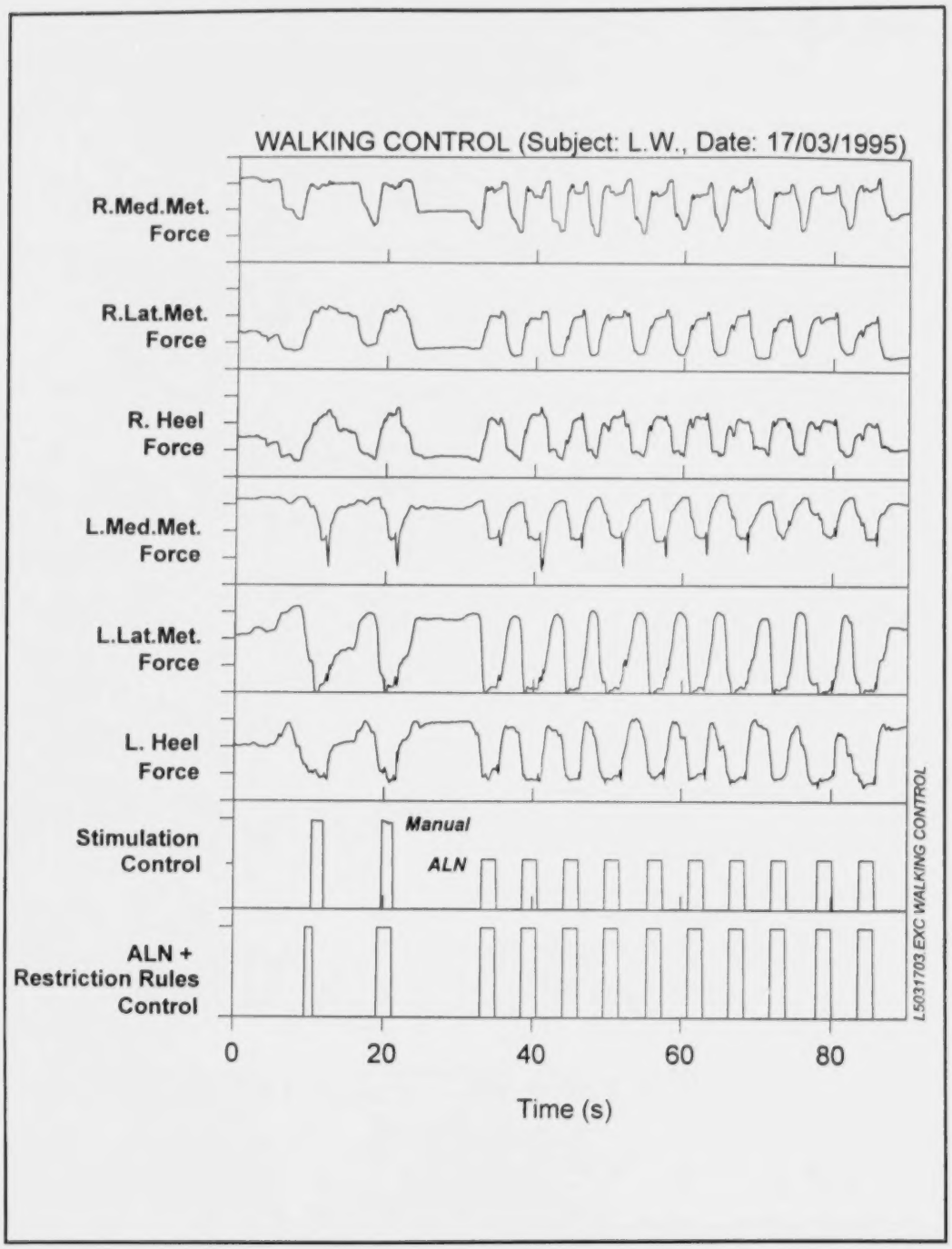

Figure 3.6.18. **Walking control** *done using ALNs trained three days before this experiment. Walking was very smooth and the subject demonstrated a full understanding of the importance of the appropriate weight transfer, as a way to voluntary inform the controller about her intentions.*

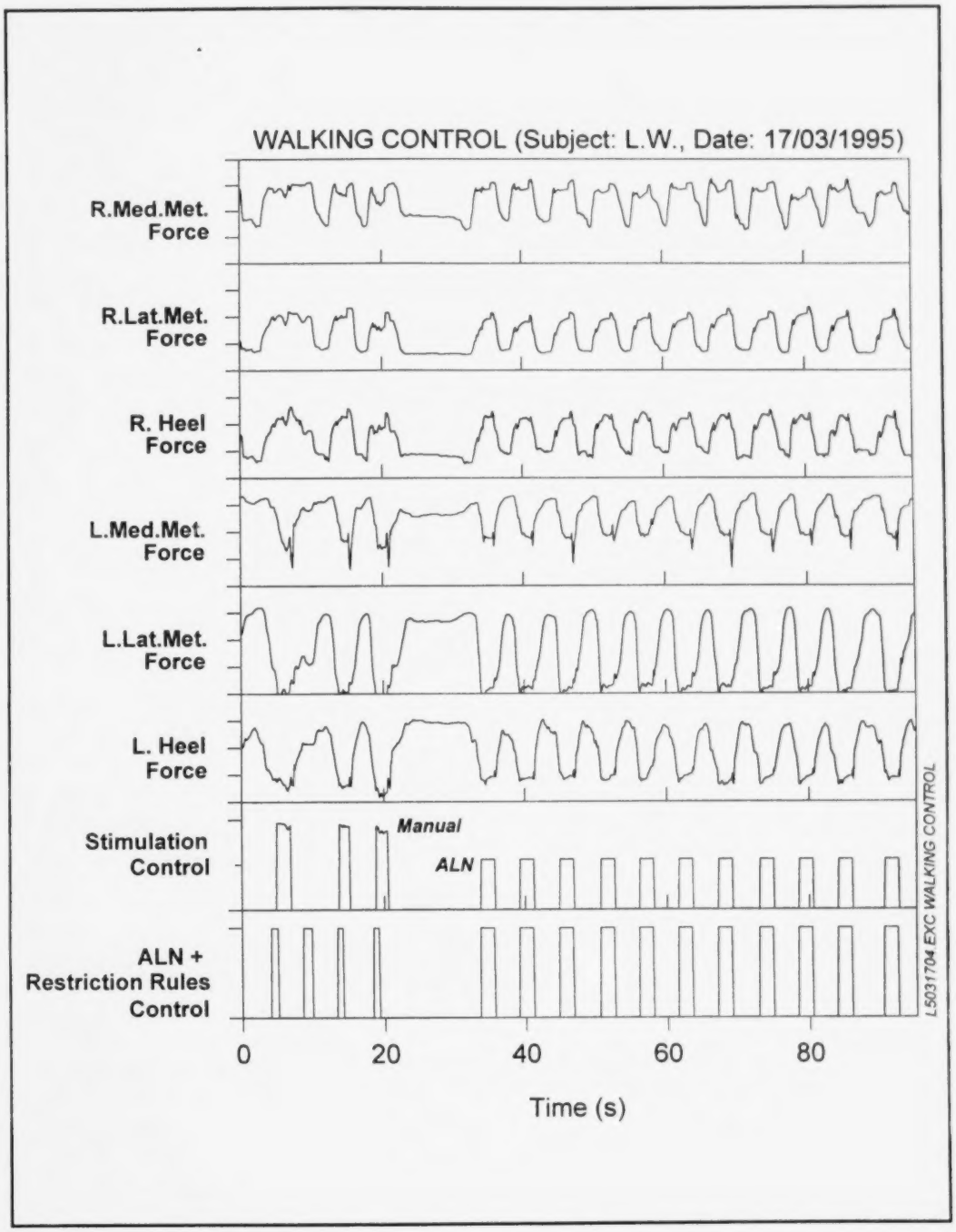

Figure 3.6.19. The second **Walking control** *round started with small instability resulting in one early and one extra stimulus (similar to the one shown in Figure 3.6.16). After the ALNs were put in charge of controlling the stimulation, the walking became as nice and smooth as one in the previous walking round.*

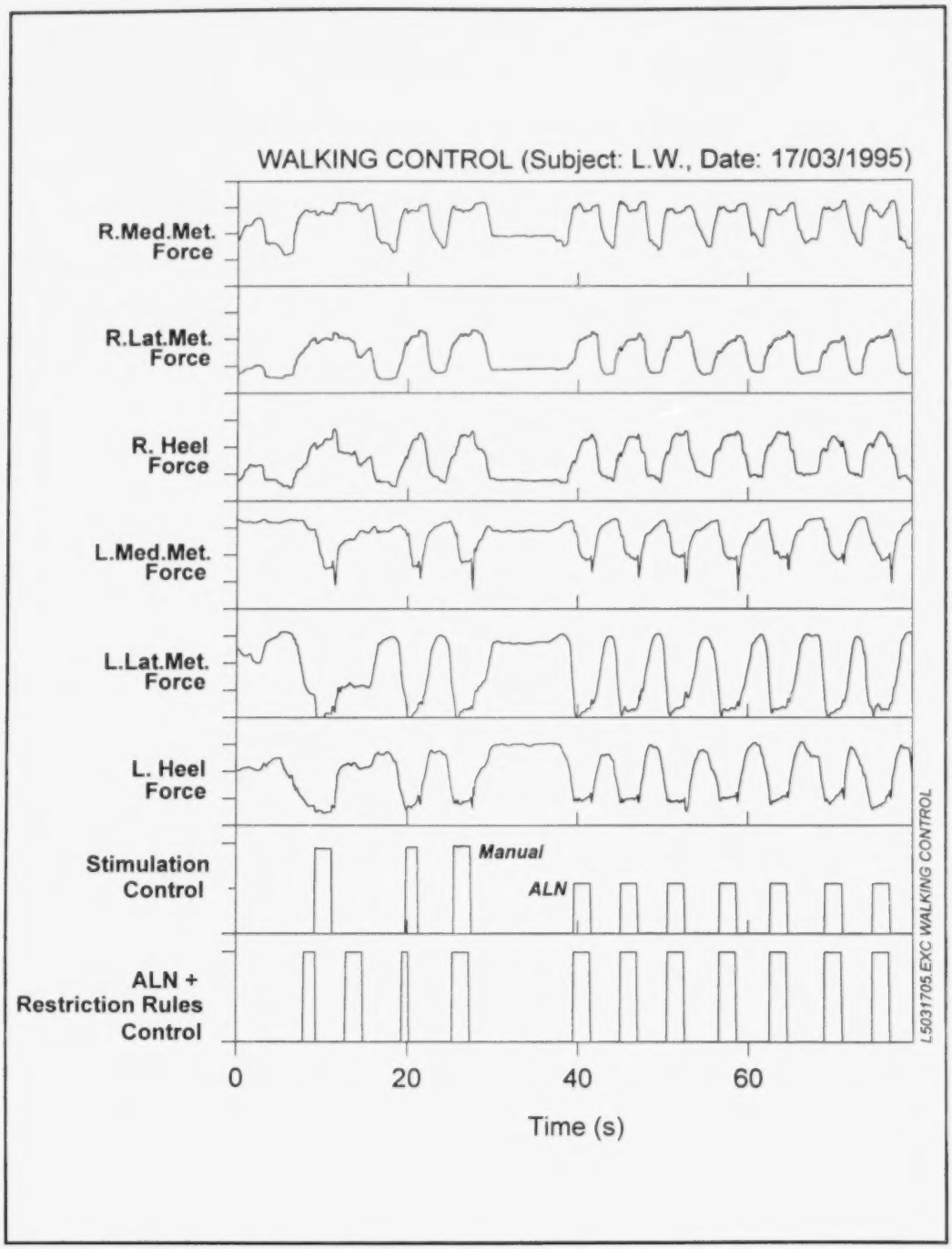

Figure 3.6.20. The third **Walking control** *round suffered the same problem at the beginning as the previous one, and it was much shorter due to development of fatigue. Otherwise, the steps recorded were regular and periodic.*

3.7 EVALUATION OF ALNs FOR CONTINUOUS CONTROL OF FES USING BIOLOGICAL FEEDBACK SIGNALS

In previous sections machine learning techniques were evaluated for control tasks requiring simple switching control outputs. It was shown that both adaptive logic networks and inductive learning technique are capable of generating control rules from samples of sensory signals, saving them in form of decision trees, and later producing control signals.

The focus of the following study was on two qualitatively new issues, as compared to the previous work: the use of natural signal sources (neural signals) for sensory feedback, and the generation of the continuous control outputs equivalent to those of natural motor drives (electromyographic signals).

3.7.1 USING BIOLOGICAL FEEDBACK SIGNALS FOR SWITCHING FES-CONTROL[14]

Among several applications that can be envisioned with natural sensors, one is of particular interest. Control of hand functions in tetraplegic subjects may be improved if a closed-loop control is implemented. Use of artificial sensors is almost impossible because of the difficulties in their installation and space limitations. Neural recordings may be very useful to estimate position on the basis of muscle spindle activity of nonstimulated muscle. In this case nerves in the forearm should be instrumented with cuff-electrodes.

---

[14] A version of this section was published in IEEE Trans.Biomed.Eng. as a part of the article "Sensory Nerve Recording for Closed-Loop Control to Restore Motor Functions", by Popovic et al. (1993). My contribution to this publication was the conceptual and experimental work related to the application of adaptive logic networks to the control problem and production of corresponding parts of the manuscript.

Another immediate application is in stroke subjects having an implant on the common peroneal nerve to enhance dorsiflexion during the swing phase of FES-assisted gait. The sural nerve can be instrumented with a cuff electrode, and recordings of the neural activity used as a trigger to start the stimulation. An obvious advantage of this technique over those available is that such a system will not require an external insole switch, force transducer or inclinometer. The use of cuff electrodes could increase the complexity of the system, so it may be necessary to include an effective telemetry system to avoid leads penetrating the skin and the always problematical connectors.

3.7.1.1 Methods

Sensory signals were recorded from TI and SP nerves using cuff electrodes and EMG activity was recorded from MG and TA muscles using epimysial electrodes. This technique was not ready for use on the human subjects and therefore the following experiment was performed on adult cats of either sex that showed suitable gait performance on a treadmill after appropriate training in a range of walking speeds from 0.4 - 1.0 m/s.

Surgical Procedure is described in detail in Section 2.2.

Monitoring: Compound action potentials(CAPs) were elicited by stimulation of nerves at 1 Hz with a pulse width of 10 μs, and an amplitude sufficient to have a maximal potential. These measurements were done for all implanted electrodes to verify that normal conduction was preserved and no nerve block occurred as a result of surgical intervention. An increase of peak-to-peak amplitude was often observed in the first few weeks after surgery in parallel with an increase in the electrode impedance (Stein et al., 1978), due to the replacement of saline with connective tissue of higher impedance. Impedances were measured with a Hewlett-Packard vector impedance meter (model 4800A) at 10 Hz, since the major component at that frequency is thermal resistance. As expected, phase angles at that frequency were typically small (< 30°). No

change in conduction velocity was observed, indicating that the nerve diameter had not changed. Neural recordings were stable, and with time, less EMG contamination was recorded (see Figure 3.7.1). Results from three cats for more than 200 days showed that the relative amplitude of the peak to peak CAP remained constant. In comparison the relative contamination of the CAP by EMG decreased somewhat in all chronic cats.

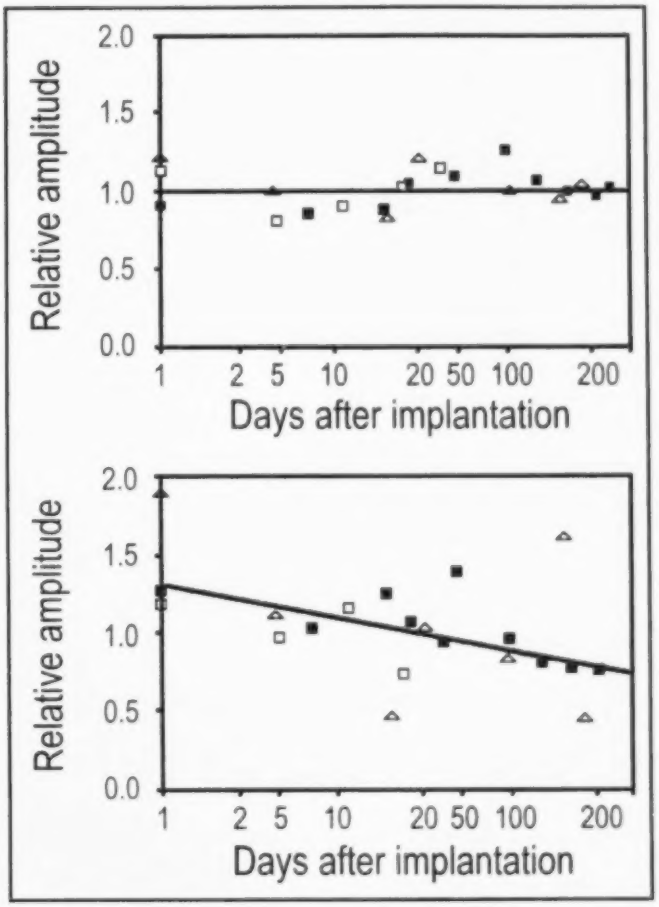

Figure 3.7.1. The amplitudes (top) of compound action potentials were measured with triphasic cuff electrodes on three different nerves: sciatic, tibial and superficial peroneal in three legs (different symbols) of two chronic cats. The potentials were elicited by stimulating the sciatic nerve and recording from the other two nerves which are branches of it and by stimulating the branches while recording from the sciatic nerve. The four values relative to the mean for each nerve over the entire recording period were averaged and plotted. The amplitudes of EMG contamination on the other nerves (bottom) were also averaged in the same way when the sciatic nerve was stimulated maximally. The EMG contamination could easily be distinguished because of its greater latency and slower time course.

Signal Processing: Neural recordings in peripheral nerves elicited from cutaneous receptors or muscle spindles are typically in the range of 3 to 10 μV. The frequency spectrum has a peak around 2 kHz with most power concentrated between 1 and 5 kHz (see Figure 3.7.2). The thermal noise with nerve electrodes is in range of 1 μV, comparable in size with the neural signals.

Peripheral nerves are surrounded by muscles producing electrical signals that are orders of magnitude larger, but have a peak near 200 Hz (Figure 3.7.2.(a)) and most power below 1 kHz. This signal is partially suppressed using a good fitting cuff electrode and suturing it tightly shut. In addition, the triphasic nature of the recording produces additional common mode rejection (Stein et al., 1975). Nonetheless, for the tibial nerve near the ankle, two distinct peaks are observed in the spectrum (Figure 3.7.2.(b)), but high pass filtering at 1000 Hz (fourth order Krohn-Hite filter, model 3700) can eliminate virtually all EMG contamination (Figure 3.7.2.(c)). With other nerves (SP, sural) the location near major muscle groups may require a cutoff frequency of 1 500 or even 2000 Hz or more.

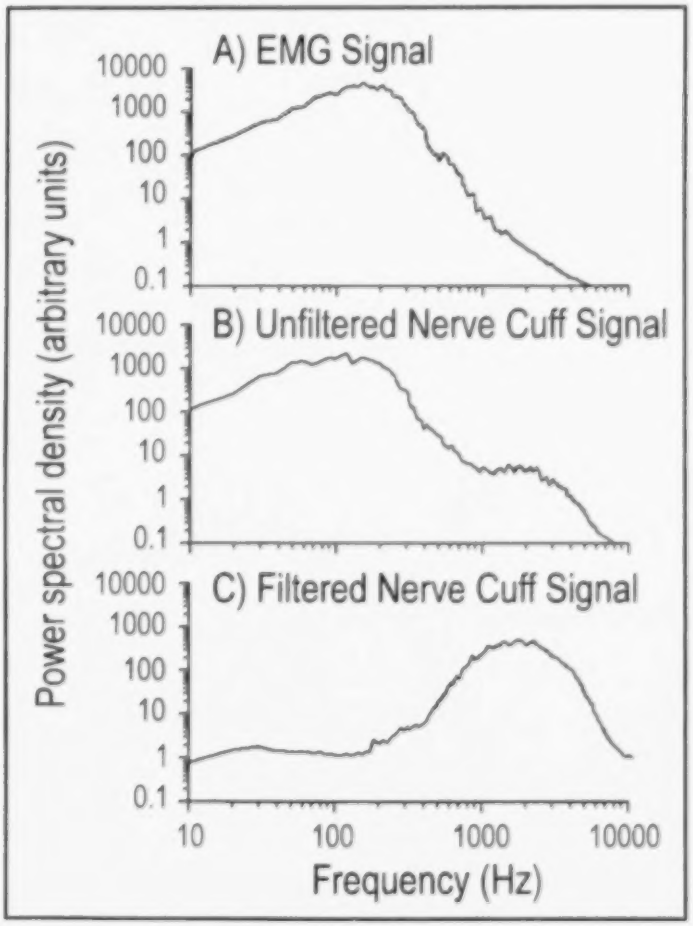

Figure 3.7.2. (a) Power spectral density calculated from EMG recorded from medial gastrocnemius (MG) muscle, while a chronic cat walked on a treadmill. (b) Corresponding spectra are shown from a nerve cuff on the tibial nerve without filtering. (c) After high-pass filtering with a cutoff at 1000 Hz. Note that without filtering the peak spectral density of the EMG is approximately 100 times that of the neural signal (10 times the amplitude), but the ratio can be reversed using a fourth order high-pass filter.

Circuit Design: In FES applications stimulation produces a large signal which is usually considered as an artefact and which is best eliminated by using a blanking circuit (Hoffer and Haugland, 1992). To measure the average output from the nerve, the signal is rectified and low-pass filtered, using a combination of RC and Paynter filters. Although these techniques have been used for a number of years with rack-mounted equipment, the challenge was to develop a portable system small in size, weight and power consumption. In addition, it should be easily integrated into the FES system, which consists of the following cascaded blocks: low noise and low input impedance preamplifier, bandpass filter, amplifier, blanking circuit, rectifier, Paynter filter, controller, and stimulator. The preamplifier was made for nerve recordings using INA110 Burr Brown instrumentation amplifier. To increase the signal to noise ratio, and to match the low impedance typically observed with nerve cuff electrodes to the high impedance that is suitable for FET preamplifiers, a miniature step-up transformer (turns ratio of 20) was employed (PICO 24400). An INA110 instrumentation amplifier was set to have the gain of 200, which made the total gain of the preamplifier equal to 4000.

The output of the INA110 is fed to the fourth-order bandpass filter and then to a two stage amplifier using precision operational amplifier (OP77) with selected gains of 75, 400, 1500, and 2250 times 10^3. Total amplification was in the order of 10^4 for EMG signals and 10^6 for nerve signals. The blanking circuit (realized with two 14066 CMOS analog switches) grounds the amplifier just before the stimulation and releases it after the stimulation. To prevent the output from decaying, the low-pass filtered signal is sampled and held for the period of the blanking. This blanking circuit is transparent for the signal and allows recordings whenever the stimulation is turned OFF.

Rectification is accomplished with a full wave rectifier. Smoothing of the rectified signal effectively creates a signal similar to the envelope of the neural or muscular electrical activity. This signal can be amplified and offset. The controller can be either a microprocessor-based circuit or a preprogrammed discrete circuit. The simple control for the stimulation in this project uses fixed

thresholds; thus, the simplest controller can be reduced to a comparator which triggers a biphasic, charge-balanced stimulator.

CMOS technology was used widely to reduce power consumption. The total power consumption is currently in the order of 9.5 mW and the conventional printed circuit boards occupy a volume of 8 cm^3 per channel. Special attention was given to the layout of components and realization of ground loops to maximize signal to noise ration. Further details can be found in Nikolic and Popovic (1992) and Nikolic et al. (1994).

Rule Based Control of the FES System: The control architecture system adopted in the present system is based on the finite-state approach to the control of prosthetic and orthotic devices. The implementation of the finite-state approach is based on the use of rule-based control. Rule-based methods belong to nonanalytical control systems and have an IF(...) THEN(...) ELSE(...) structure (Andrews and Bajd, 1984; Popovic et al., 1991). The cyclic motor activity is presented as a sequence of discrete events. Each of these discrete events is associated with a unique sensory pattern. A sensory pattern occurring during particular motor activity is recognized with the use of artificial and/or natural sensors. The specific discrete event is called the state of the system in analogy to the state of finite-state automata. A recognized sensory pattern during a specified state of the system initiates corresponding functional movement.

Rules in a rule-based control system were initially hand-crafted. This involved human expertise and was very appropriate for simple systems having a limited number of states. The expertise for detecting rules can be classified as a pattern recognition skill. An alternative to hand-crafting event detection rules is to use rule induction methods, developed for machine learning in artificial intelligence (Andrews et al., 1989; and Veltink et al., 1990). The development of artificial neural networks provides a new tool for computer generation of rules for control of an assistive device.

Adaptive Logic Networks: If the transfer function between input and output of a system is not known and cannot be easily described in an analytical form, if at all, artificial neural networks can

Adaptive Logic Networks: If the transfer function between input and output of a system is not known and cannot be easily described in an analytical form, if at all, artificial neural networks can be used for approximation of the required mapping. One of the interesting control problems in design of a prosthesis is to map the relationship between afferent neural signals and activation of muscles of the same or the opposite limb. We have attempted to reproduce this mapping using adaptive logic networks. In our experiment the inputs to an ALN were derived from neural recordings presented in the first two traces of Figure 3.7.3. (tibial nerve and superficial peroneal nerve). Each input signal was quantized to a certain number of levels and each level was converted into a boolean vector in such a way that close levels resulted in vectors which were close in Hamming distance (Smith and Stanford, 1990; Armstrong et al., 1991). The required output of the system was boolean (trace 2 in Figure 3.7.6), namely a thresholded EMG signal from trace 1 in Figure 3.7.6 (medial gastrocnemius EMG). The output signal has been considered as an indication of the muscle activity.

Version of the ALN learning program: **ALN V.2** (see section 2.7.1 and Appendix B for details).

ALN interface program: **WALKON** (see section 2.8.1 and Appendix C for details).

Encoding technique: **Random Walk and Unary Encoding** (see section 2.7.1.1).

3.7.1.2 Results

We concentrated in this study on controlling a single joint by stimulating ankle extensors and flexors and processing signals using custom built electronic circuits (see Methods). Figure 3.7.3 shows typical recordings from the SP and tibial nerves as well as ankle extensor (MG) and flexor (TA) muscles of a chronic cat. The signals cover a period of 20 s, where the cat stood for 3 s, made one step, stood again for about 2 s, climbed up on the wall of the treadmill with its forepaws while balancing on its hindlegs (note the strong burst of activity recorded in the MG muscle), took another step and then stood for 2 s before starting to walk rhythmically on the treadmill with a

cycle time of ≈ 750 ms. There is a characteristic double burst with each step in the SP nerve and the single burst in the tibial nerve. Furthermore, although the signal amplitudes are much smaller, the processed electroneurograms (ENG) are much more reproducible in amplitude from step to step than the corresponding EMG signals.

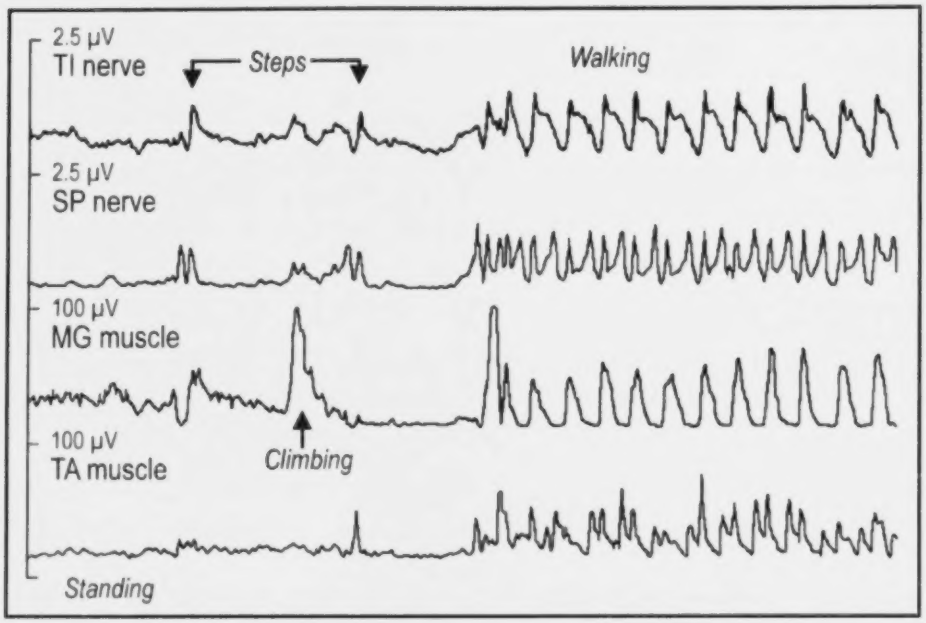

Figure 3.7.3. Recordings from superficial peroneal (SP) and tibial (Tib) nerves as well as medial gastrocnemius (MG) and tibialis anterior (TA) muscles of a chronic cat. This 20 s recordings were selected to indicate the distinct rhythmic activity in peripheral nerves and ankle muscles in various behaviors.

The ENG signals were also used to design a simple state diagram, based on setting thresholds on the filtered nerve records (dashed lines on Figure 3.7.4). The single peak in tibial nerve activity occurs each time the cat' s paw hits the ground and corresponds closely to the time when the MG activity begins. Thus, the first rule in Figure 3.7.4 is:

> *if* the tibial nerve signal exceeds threshold,
>
> *then* activate the MG stimulus for a period corresponding to the average duration (307 ms) that the MG activity remains above a threshold level (not shown).

As mentioned above, the SP nerve shows two peaks, one corresponding to the time when the paw hits the ground and another, larger peak when the paw is lifted off the ground. The latter peak corresponds to the onset of the TA activity. Thus the second rule is:

> *if* the SP activity exceeds its threshold and the tibial nerve is below its threshold,
>
> *then* activate the TA stimulus for a period equal to the average duration (420 ms) that the TA activity remains above a threshold level.

These two simple rules are sufficient to reproduce the basic alternating pattern between the flexors and extensors controlling the cat's ankle joint.

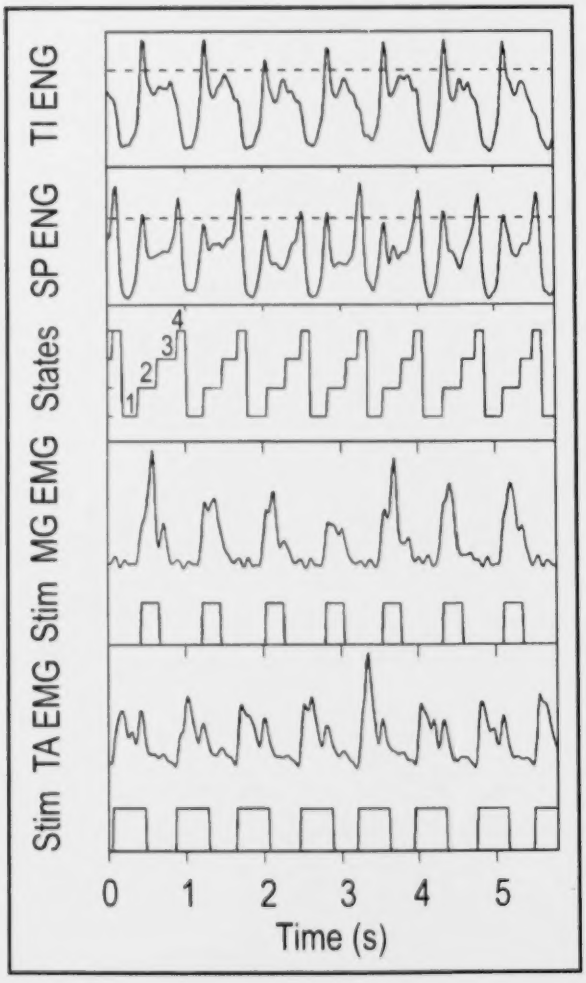

Figure 3.7.4. Threshold levels were set (dashed lines) to determine instants when processed electroneurograms (ENG) recordings from tibial and superficial peroneal nerves (upper two traces) crossed above these preset levels. Threshold crossings activated transitions in a simple rule-base with four states (third trace), as explained further in the text. Entering state 2 turned on circuits for a fixed period of time (trace 5) that correspond on average to the duration of suprathreshold activity of MG EMG (trace 4). Similarly, entering state 4 turned on a circuit for a period (bottom trace) corresponding to the suprathreshold period of the TA EMG (trace 6). Note that the circuit operated reliably with no false positive or negatives.

In the example shown in Figure 3.7.3, no stimuli were actually applied. However, with the blanking circuit it was possible to stimulate the muscle to generate substantial force without affecting the sensory nerve recordings. Recordings from the tibial nerve for one gait cycle are shown in Figure 3.7.5, and for tibial and superficial peroneal nerves when the cat is walking in Figure 3.7.4. Hoffer et al. (1989) obtained similar results but used a computer to process the signals. A number of years ago Hoshimiya et all. (1976) developed an analog circuit for this purpose. The results shown here utilized a small battery operated circuit that a subject could use practically on a daily basis.

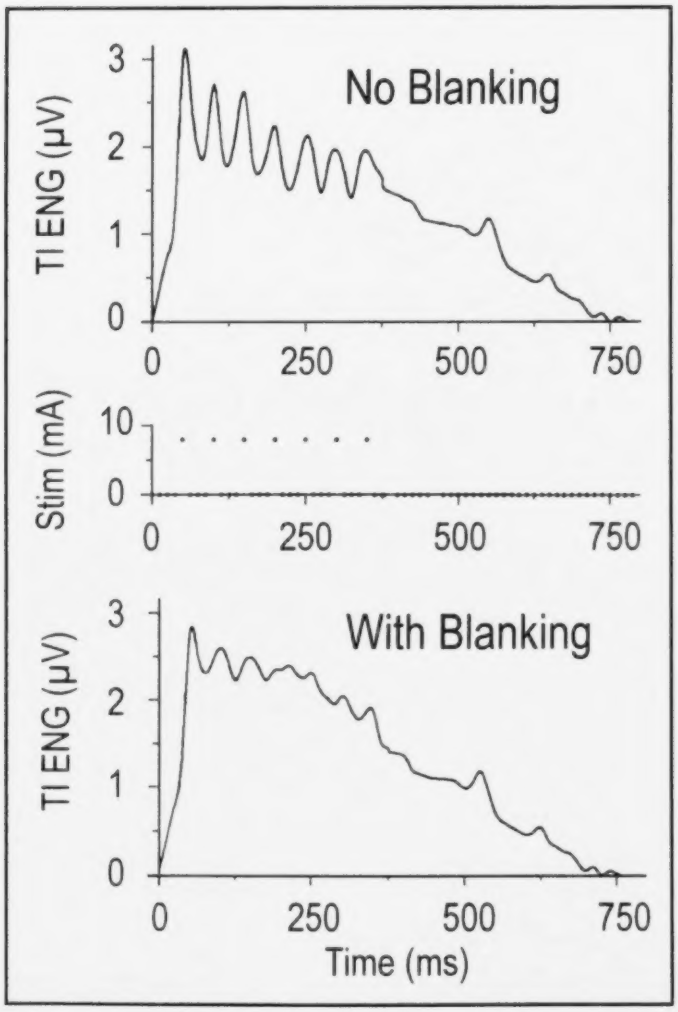

Figure 3.7.5. When the rule-base was connected to a stimulator and stimuli were applied (middle trace), artifacts were generated on the processed tibial ENG (top trace), even with filtering, but this could be greatly reduced (bottom trace) by switching in a blanking circuit.

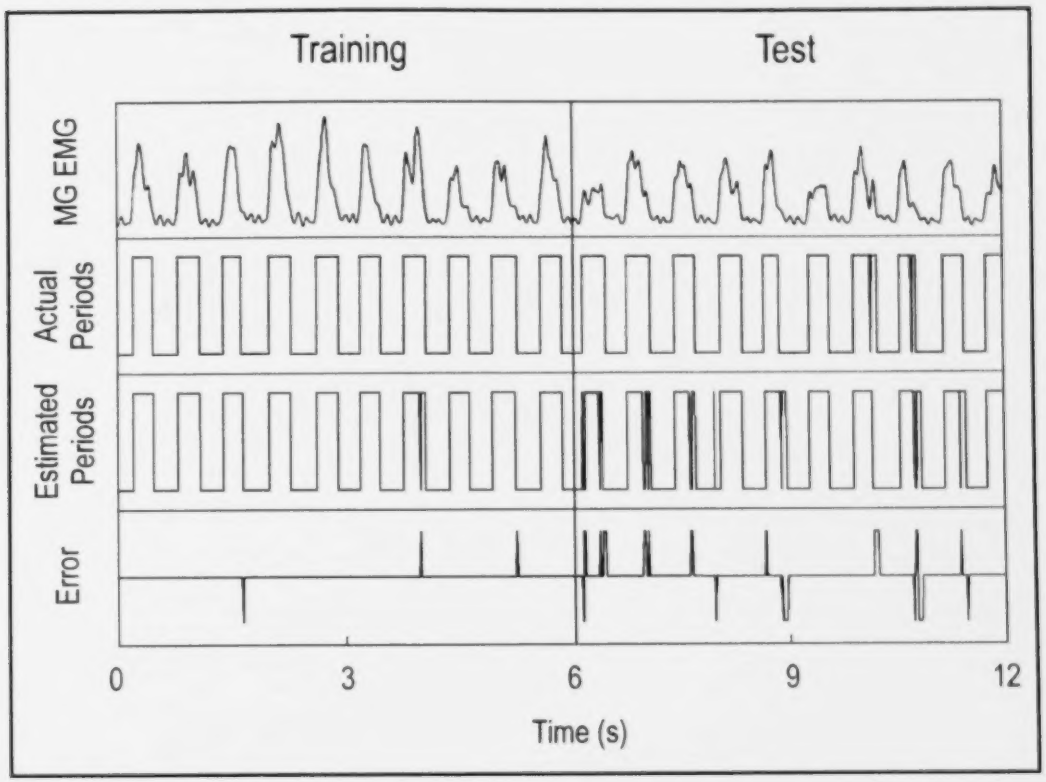

Figure 3.7.6. To match the activity pattern of an ankle muscle, based on the signals from the tibial and SP nerves, we used an adaptive logic networks (ALN). Neural signals (upper two traces in Figure 3.7.4) were presented to the ALN as the input, together with an output which consisted of pulses (second trace in this figure) representing periods when MG EMG exceeded a threshold value. The data on the left half of the figure were used in training the ALN to reconstruct the binary representation of the EMG signal. The reconstructed result is in the third trace, and the error is in the fourth trace. The right half of the figure presents test data, where reconstruction was not as successful as in training. Further details in the text.

The rule base of Figure 3.7.4 only matches two features of the normal EMG signals, their onset and average duration, but it does so reliably (no false positives or false negatives) if input signals (neural recordings) are of a certain amplitude and shape as described earlier. It may be desirable for some purposes to match the exact timing for each step profile. We examined if adaptive logic networks could perform this function in two series of trials. The first served to find the number of samples from the past required for training and an optimal region in the prediction error plane. The second produced the actual results of training in that region. Based on previous experience with

similar signals (Kostov et al., 1992), a decision was made to quantize neural signals to 128 levels and encode them with 14 bits. In contrast, the EMG signals were quantized to 2 levels and encoded with only 1 bit. The best training results were obtained using the original neural signals plus two copies of the same signals delayed 80 and 160 ms and two copies of differentiated neural signals with the same delays. The artificial neural trees contained 1536 nodes, the majority vote of seven trees being taken to compute an output bit, the number of presentation (epochs) of the training set to the learning algorithm during training of every tree was 20. The results from the ALN are presented in Figure 3.7.6. The top trace is the EMG signal. The second trace shows the actual intervals when EMG activity was above the threshold. This binary representation of the MG EMG was presented as a target signal to the ALN during the training. The neural recordings, shown in the first two traces in Figure 3.7.4, were used as input to the ALN representing a domain. The third trace presents the reconstruction of the actual EMG intervals by the ALNs. To compare the actual timing with that predicted by the ALNs, we included the errors in the bottom trace. The errors during training were very few (below 1 %, left half of Figure 3.7.6), but increased considerably when we tested the performance of the ALN against data that it had not seen before (over 8 %, right half). This error can be reduced if different filtering is applied to the original signals. However, the present paper is not intended to be a test of neural networks but merely to demonstrate that ALNs applied to neural and muscular signals may be appropriate for determination of invariant features in the locomotion and as a replacement of hand-crafting rules for control.

3.7.1.3 Discussion and conclusion

To prove our hypothesis that multielectrode recordings from sensory nerves can provide satisfactory information for a rule-based control, we used a chronic cat model. The ankle joint of the cat's hind limb was selected for several reasons: 1) many characteristics of peripheral nerve recordings and muscle signals are well described in the literature (e.g., Stein et al., 1975 and

1977) a percutaneous system consisting of several nerve-cuff and epimysial recording electrodes can be easily installed; 3) cats can be trained to perform simple motor tasks, such as walking on a powered treadmill and 4) long term recording and stimulation has proven to be effective. In this model system recordings from two sensory nerves (TI and SP) were sufficient to trigger the appropriate periods of stimulation reliably to ankle flexor and extensor muscles. The exact timing of activity could be learned using an ALN.

3.7.2 CONTINUOUS FES-CONTROL USING BIOLOGICAL SENSORY FEEDBACK SIGNALS[15]

The study presented in the previous sections demonstrated the use of biological feedback for ON/OFF type of control of ankle flexion and extension in a chronic cat preparation. Encouraged by the success of adaptive logic networks in correctly restoring and predicting muscle activation timing from afferent recordings, in the following study we experimented with restoring and predicting the full shape of the corresponding EMG signals while a cat walked on the powered treadmill. Sucessful prediction of the EMG patterns would provide not only the time parameters of the stimulation, but also the intensity of the stimulation at every instant of the gait cycle, given the relationship between the EMG signal, intensity of the stimulation and the force produced by the stimulation. The experimental hypothesis was that cutaneous and proprioceptive sensory afferent signals carry enough information, which can be used to describe precisely simultaneous activity in ankle flexor and extensor muscles.

The experimental design involved in three steps: training, test and real-time application. Two of these three steps are performed in this study and presented in Figures 3.7.7 A) and B). The third step, actual real-time application remains to be done in future work.

---

[15]A version of this section was presented in the form of a short article "Learning of EMG-patterns by Adaptive Logic Networks", by Kostov et al. (1993), at the 15th Annual International IEEE - EMBS Conference. The article is published in the Proceedings of the Conference.

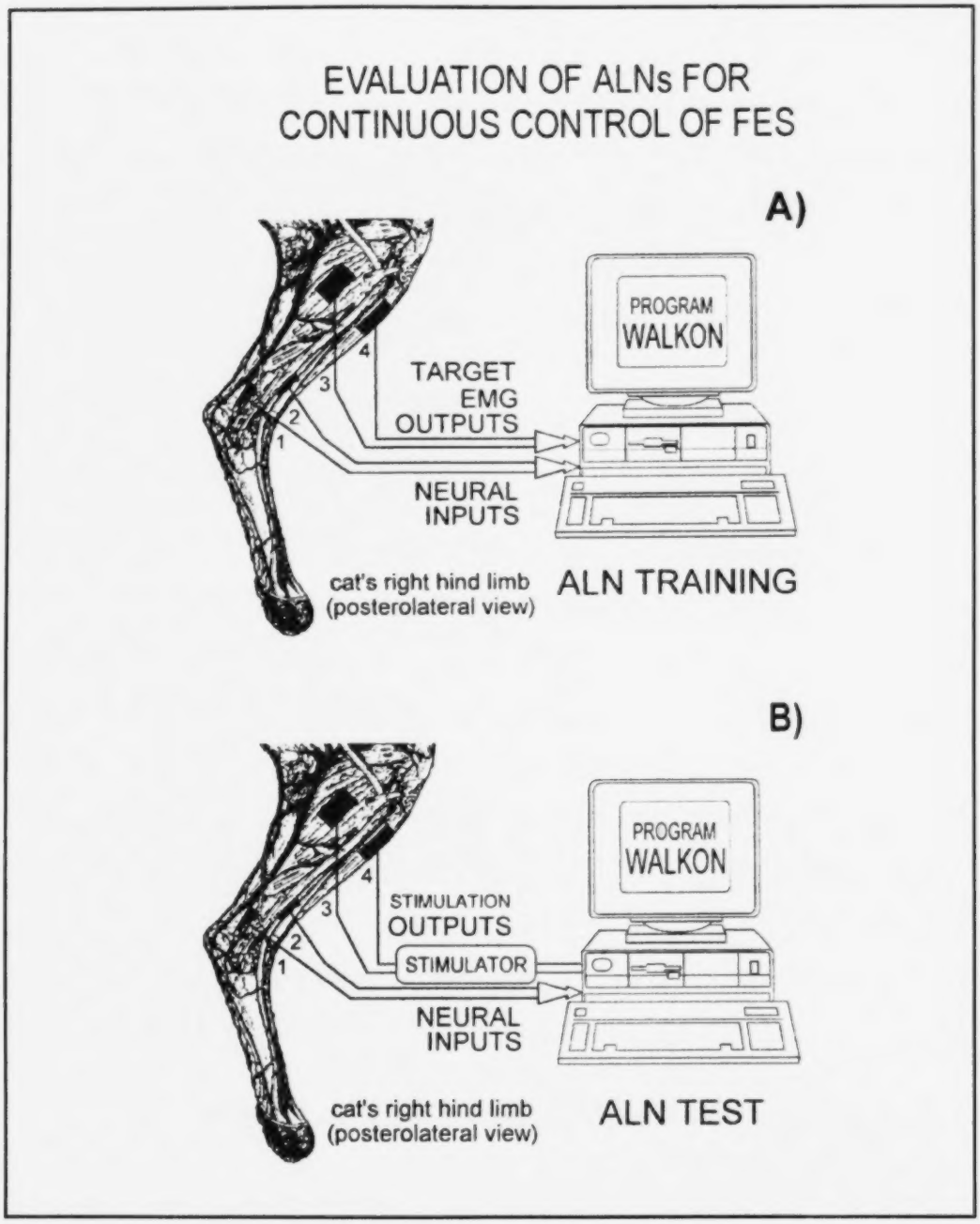

Figure 3.7.7. Configuration setup for evaluation of ALNs for continuous control of FES. A) Training of the ALNs to approximate and predict target EMG signals from neural inputs during walking on a treadmill. B) Test of the trained ALNs on neural signals not used during the training. The ALNs are here predicting the muscular activity from indirectly related neural activity.

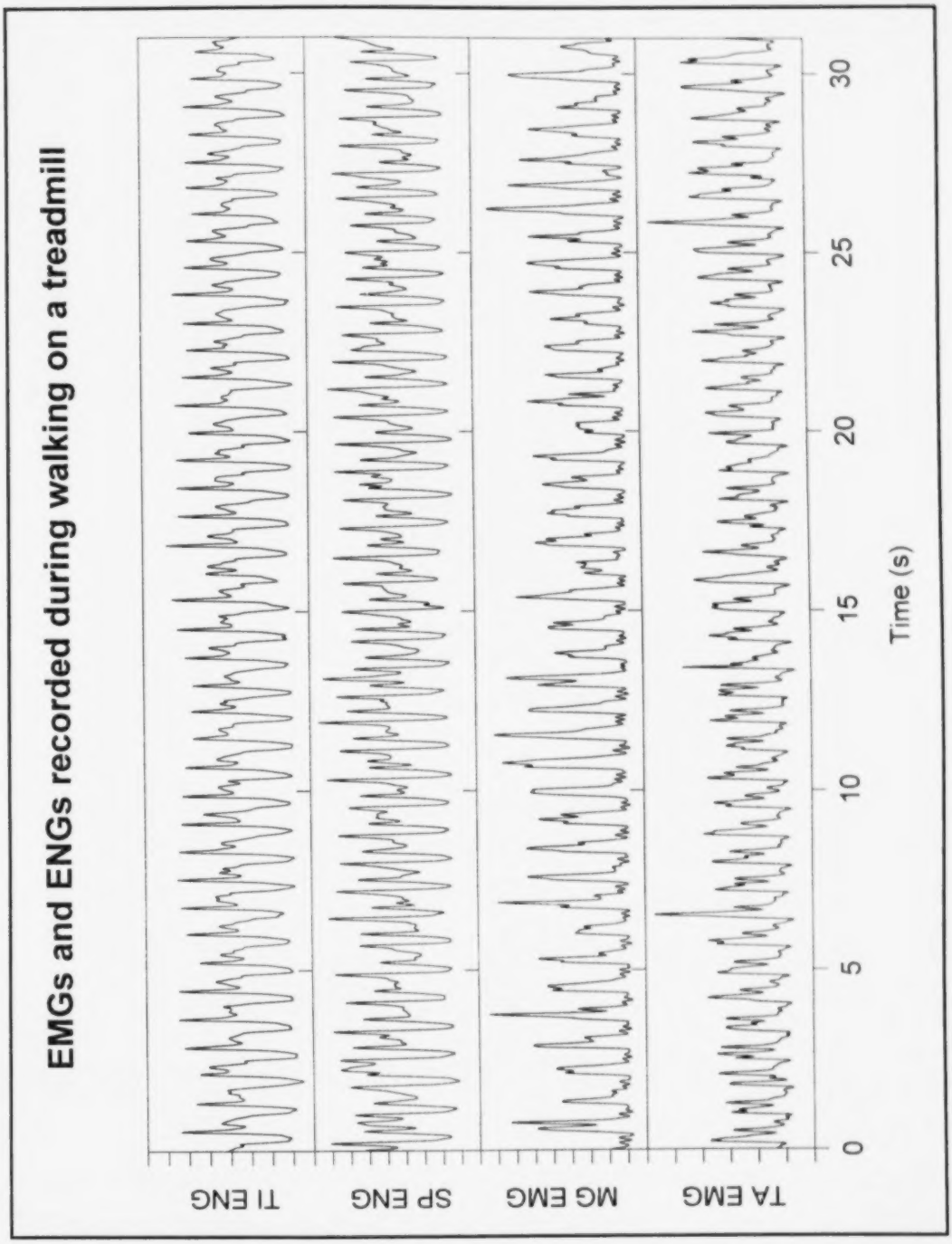

Figure 3.7.8. A sample of neural and muscular activity presented in the form of ENG and EMG signals recorded in the cat's hind limb during walking on the powered treadmill. A degree of shape irregularity is much higher in the EMGs than in ENGs.

3.7.2.1 Methods

Neural signals from the tibial (TI) and superficial peroneal (SP) nerves near the ankle of a cat's right hind limb and EMG signals from medial gastrocnemius (MG) and tibialis anterior (TA) muscles were recorded from the cat walking on the powered treadmill. The signal recording, conditioning and preprocessing was as described in Methods of the section 3.7.1. Recording resulted in data files each containing about 40 steps over about 1700 data samples (sampling rate 50 Hz). Figure 3.7.8 illustrates a sample of the signals recorded during several steps. TI ENG (electroneurogram) and SP ENG are sensory signals used as an input in the ALN algorithm, and MG EMG and AT EMG are myoelectric signals used as outputs to be reproduced by the machine learning algorithm.

Three data files were recorded and split into halves. Each half of the file was used for training, while the other half was used for testing.

Different methods were used to analyze and visualize the relationship between neural inputs and myographic outputs (see Figures 3.7.9 and 3.7.10). The final goal was to understand the possibilities of using afferent signals to predict muscular activity in the same limb.

Version of the ALN learning program: **ALN V.2** (see section 2.7.1 and Appendix B for details).

ALN interface program: **WALKON** (see section 2.8.1 and Appendix C for details).

Encoding technique: **Unary Encoding** (see section 2.7.1.1).

Machine learning technique ALN V.2 was used to generate the rules describing the relationship between sensory inputs and EMG outputs. The learning task was the most difficult one yet tried with this technique, due to the very small number of sensory inputs (two) and the complexity of the control output signals (EMGs). An extensive study was done in search of an optimal set of encoding and training parameters which will produce good test results in the shortest time.

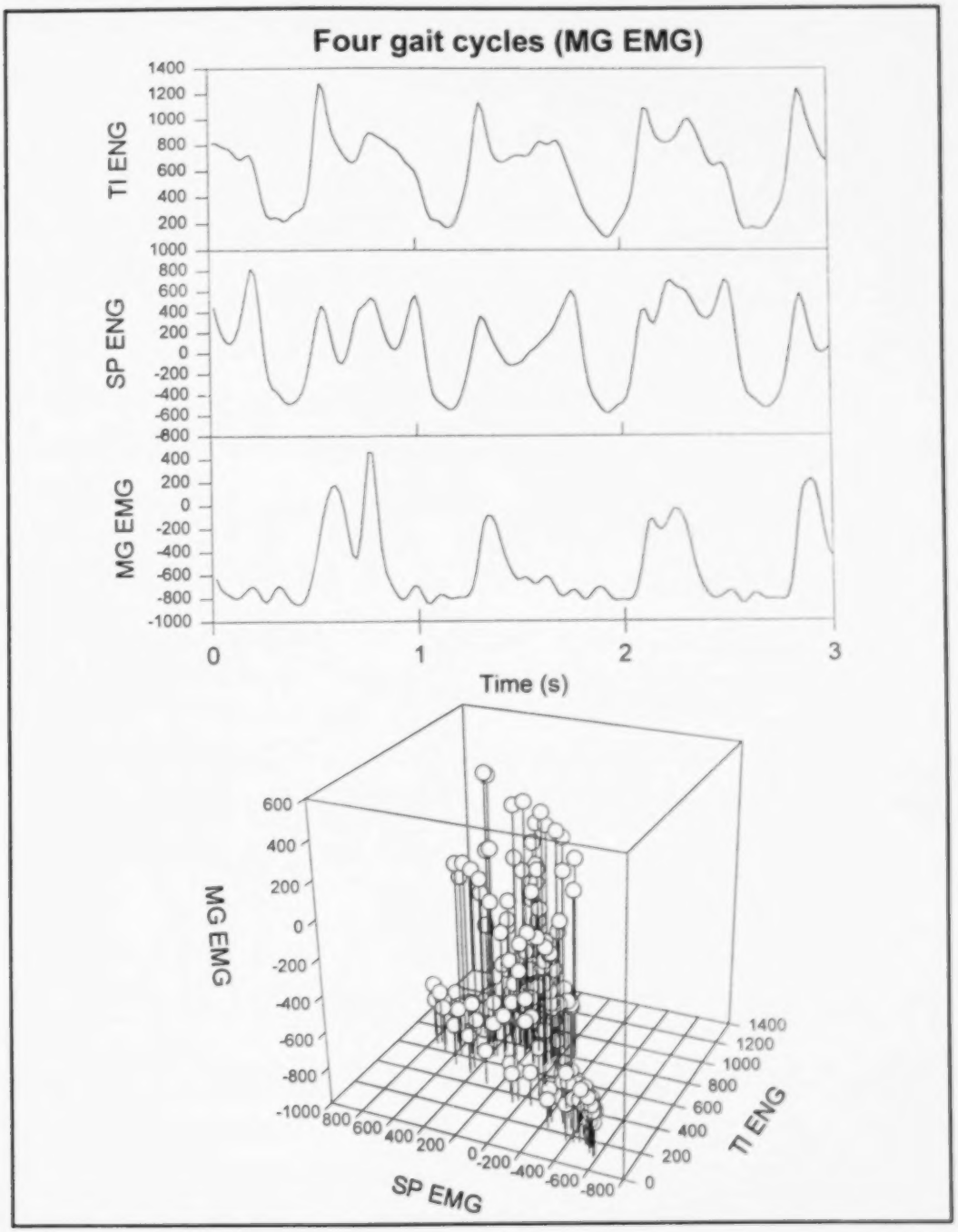

Figure 3.7.9. Three-dimensional relationship between output MG EMG signal and sensory inputs (TI and SP ENGs) in approximately four gait cycles.

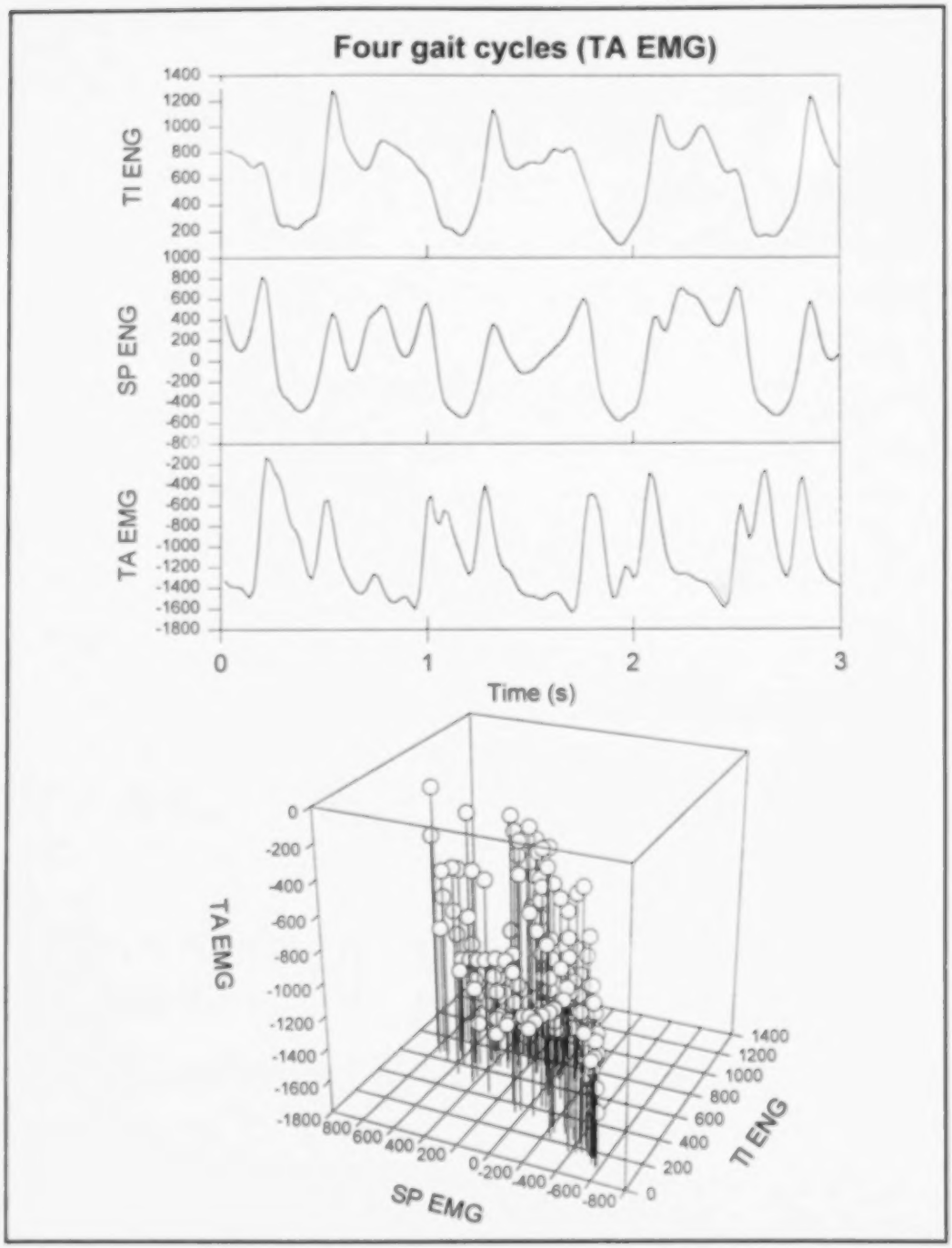

Figure 3.7.10. Three-dimensional relationship between output TA EMG signal and sensory inputs (TI and SP ENGs) in approximately four gait cycles.

The amplitude distribution for MG and TA EMGs was measured in order to experiment with the 'expert encoding', described in section 2.7.1.1. As illustrated in Figures 3.7.11 and 3.7.12, the distribution is not uniform over the amplitude range, specially for the MG EMG signal. Non-uniform amplitude distribution is a good basis for the 'expert encoding' which enables the researcher to pick those threshold levels which would produce the best representation of the signal for the smallest number of threshold levels. Keeping the number of threshold levels low provides fast training without increasing test error. This feature is built into the program WALKON which was used to do the study.

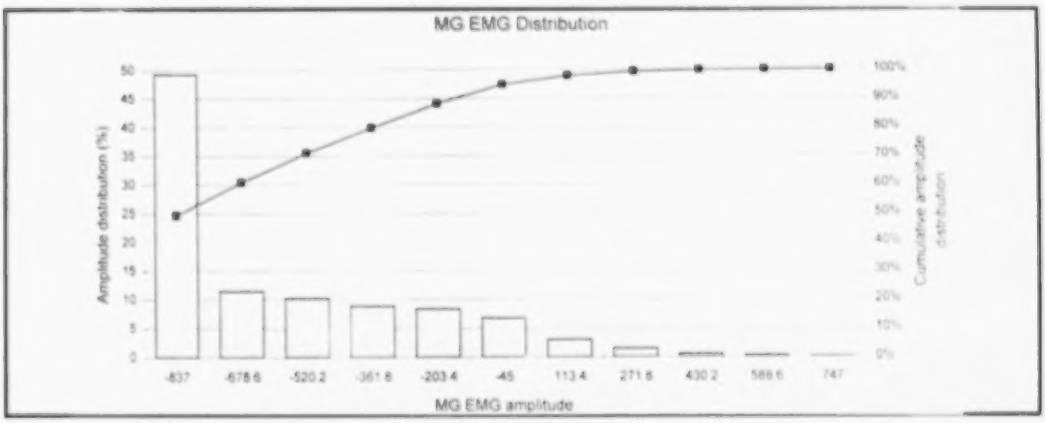

Figure 3.7.11. Amplitude distribution for the MG EMG signal.

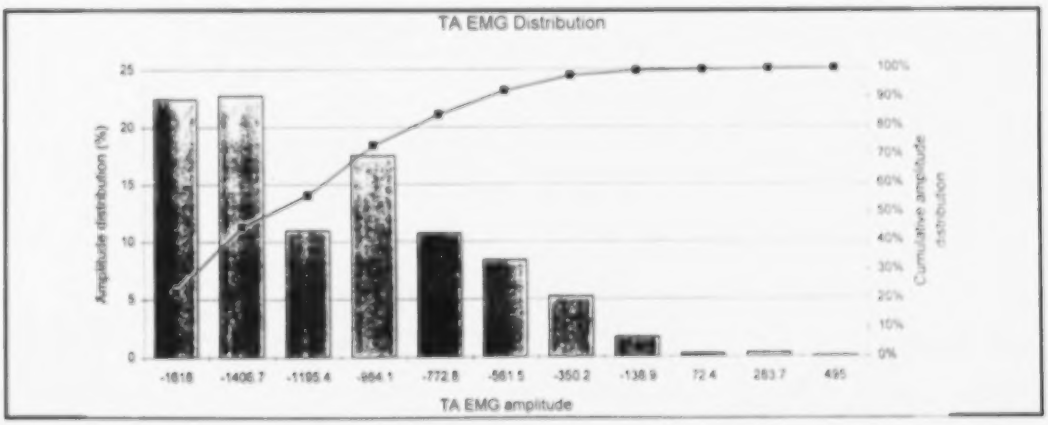

Figure 3.7.12. Amplitude distribution for the TA EMG signal.

As Figure 3.7.13 shows, 'expert encoding' resulted in significant reduction in training time, although the training and test errors obtained were not significantly affected. Since there was no change in the performance and the reproducibility of the results, we continued using uniform unary encoding instead of 'expert encoding' in further experiments. 'Expert encoding' certainly has potential, but in this particular application the achieved improvement was insignificant.

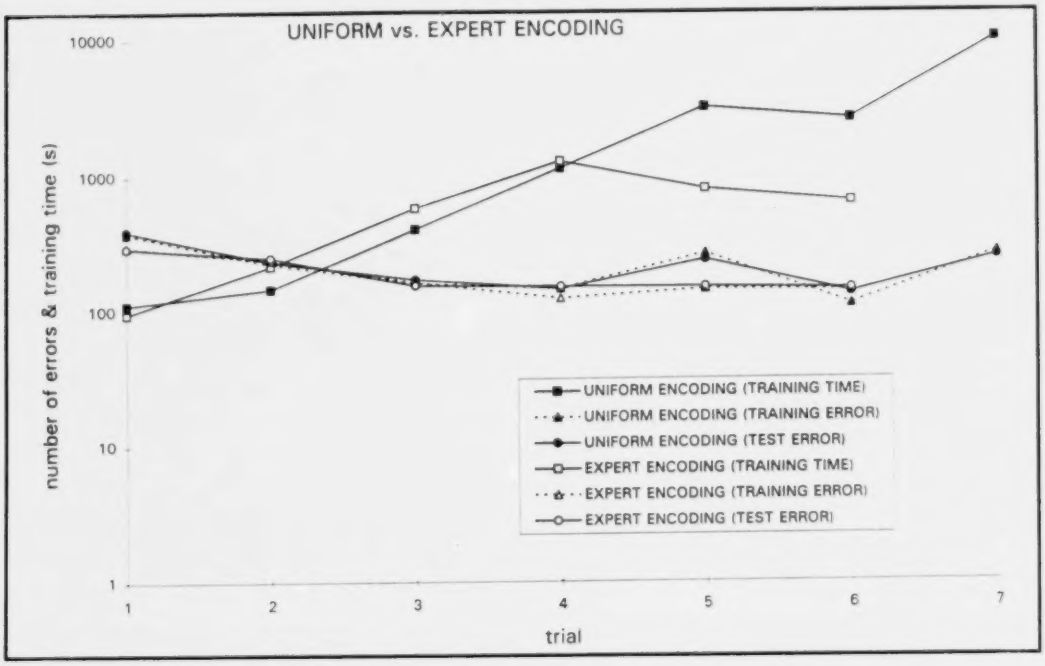

Figure 3.7.13. Expert encoding compared to uniform unary encoding.

During the process of training parameter optimization, training and test errors were defined as the numbers of those samples in which the restored and predicted output EMG amplitude respectively, differs more than 10 % (of the original EMGs range) from the original EMGs. Figures 3.7.14. and 3.7.15. illustrate the optimization of the ALN training parameters with regard to achieved training and test error respectively.

A summary of the optimisation process is presented in Figure 3.7.16. ALN training time was measured while the number of uniformly distributed unary encoding levels in MG EMG and the

size of the ALN trees were varied. It can be seen from Figure 3.7.16 (thinner lines) that by increasing the number of encoding levels from 4 to 64, training time increases. If the number of encoding levels further increases, some training sessions do not converge, probably because the ALN trees were too small for the task. Note that the first training session which starts showing divergence does not have to be the one with the smallest ALN trees, due to the random processes involved.

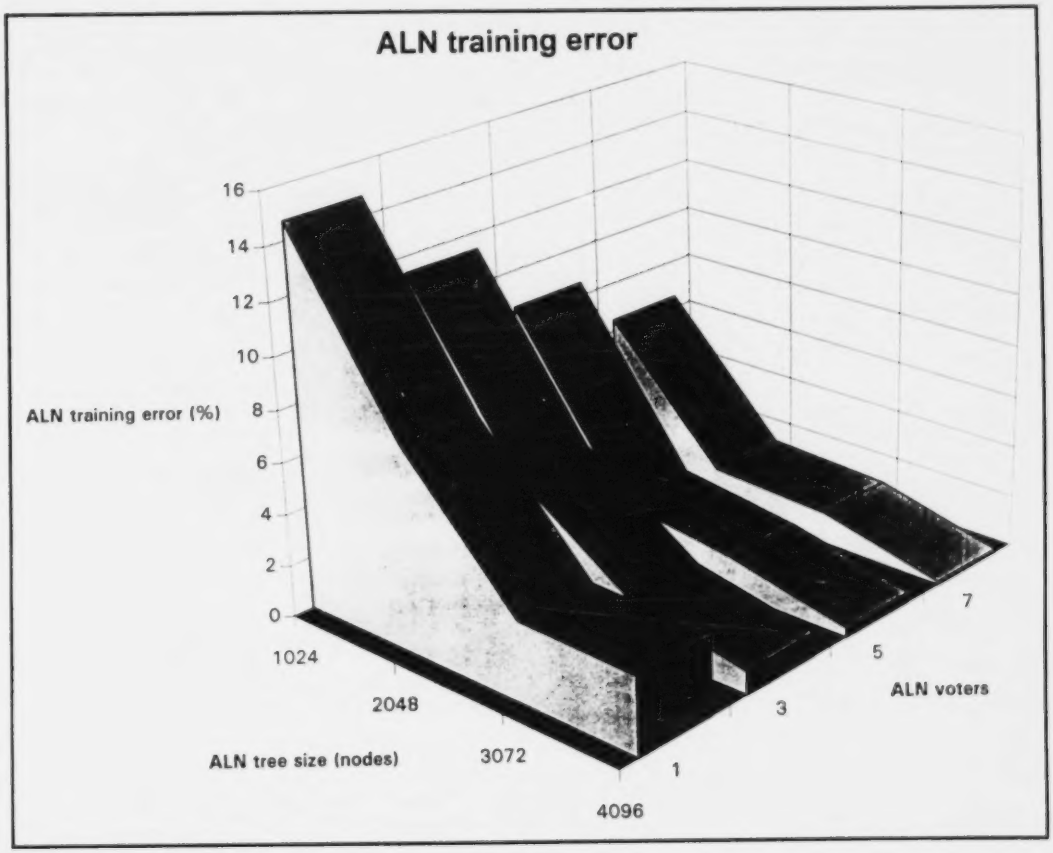

Figure 3.7.14. ALN training error

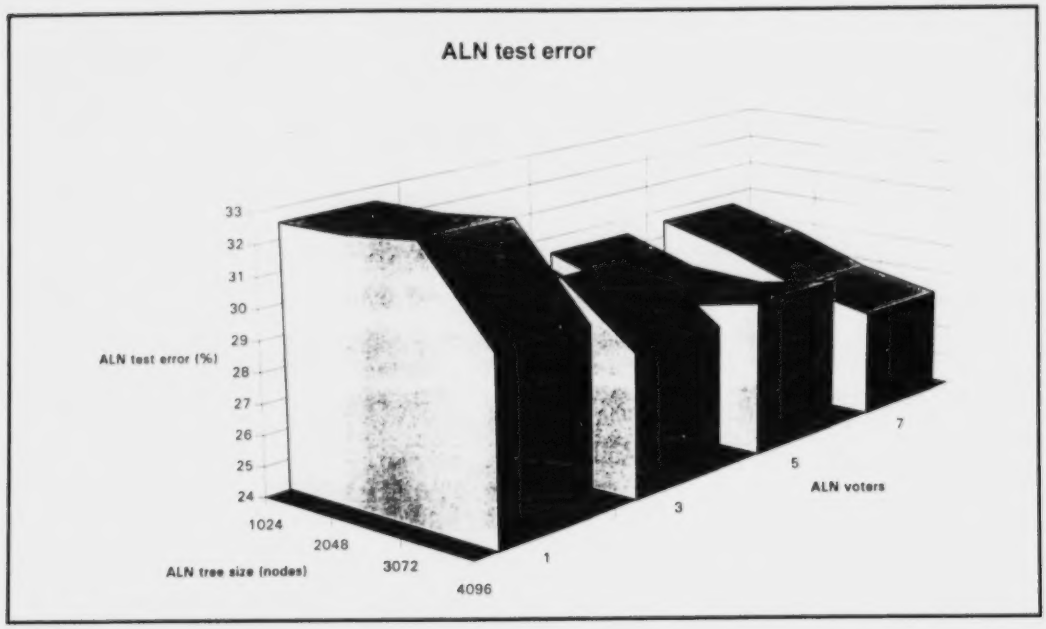

Figure 3.7.15. ALN test error

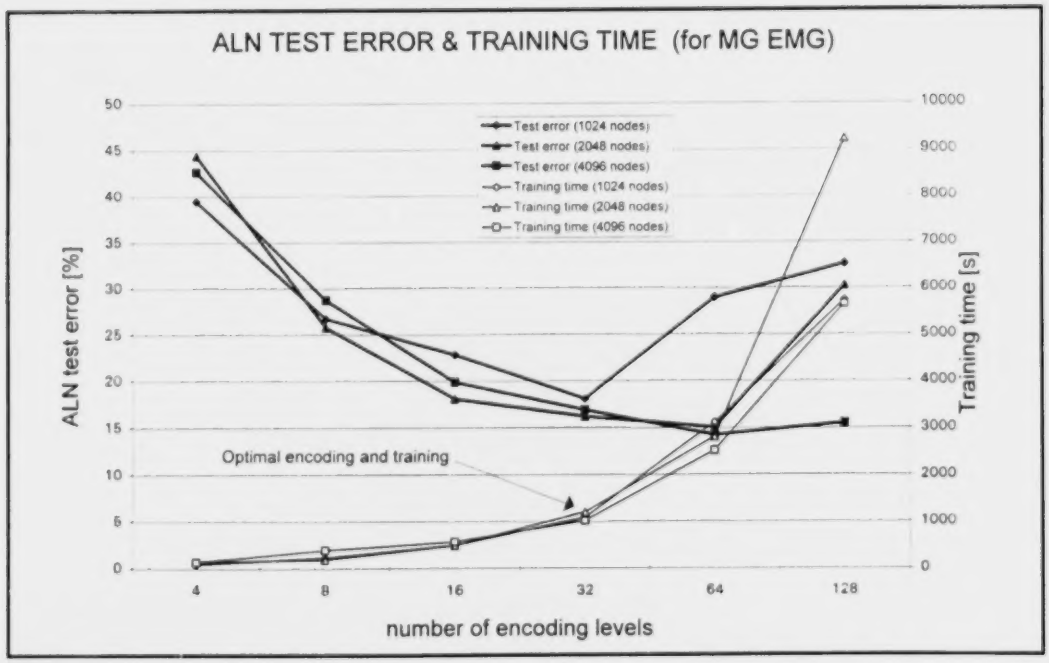

Figure 3.7.16. ALN training time and test error relationships with the number of encoding levels and the size of ALN trees.

In addition, ALN test error was measured while the number of uniformly distributed unary encoding levels in the MG EMG and the size of the ALN trees were varied. Test error was defined as the number of samples in which the predicted output EMG was more than 10% different from the original output EMG. It shows an opposite effect to the one described for training time. As the Figure 3.7.16 shows (thicker lines), increasing the number of encoding levels reduces the test error until the number of encoding levels changes from 32 to 64, when the test error increases for the smallest ALN tree size of 1024 nodes. This is a sign that the size of the ALN tree is probably not large enough to process the amount of information presented to the learning process. In support of this finding, as the number of encoding levels further increases to 128, next tree size (2048 nodes) becomes too small and only the ALN tree of 4096 nodes is still capable of learning the required task. This analysis shows a high level of predictable behavior of artificial neural networks and the possibility of intuitive designs in predicting configurations that will most successfully resolve the required tasks.

3.7.2.2 Results

The analysis described above resulted in the following encoding and ALN training setup: All signals were quantized to 32 levels and encoded with 31 bits using a 'thermometer' or unary code (described in the section 2.7.1.1). The best training results were obtained using the original neural signals together with their differentials in a configuration comprising the current sample and two samples from the past (delayed 80 and 160 ms). The size of the trained ALN trees was 4096 nodes, the number of epochs was 20 and the number of voters (ALN trees per output bit) was seven. Such a configuration required several hours to finish the training on a 486 DX/50 MHz IBM PC compatible computer.

The applied encoding and training parameters resulted in high correlation between an original and predicted EMG signals. To estimate the generalization of the learning algorithm, the variance accounted for (VAF) was calculated as follows:

$$VAF = 100\left(1 - \frac{\mathrm{var}_{fit}}{\mathrm{var}_{dat}}\right)$$

where:

$$\mathrm{var}_{fit} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \bar{x}\right)^2$$

$$\mathrm{var}_{dat} = \frac{1}{N}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2$$

x_i is a sample,

y_i is a prediction,

N is the number of samples.

Overall results are presented in Table 3.7.1. As expected, the VAF by the ALN prediction of the test data was lower than the VAF of the training data.

Table 3.7.1. VAF as a measure of ALN learning performance and generalization on test data.

	TRAINING		TEST	
	MG EMG	TA EMG	MG EMG	TA EMG
VAF (%)	95.2	94.6	79.8	80.7

An example of the restored and predicted EMG signals, together with the input sensory signals, is presented in Figure 3.7.17. The colored areas in the last two traces represent the difference between original and predicted EMGs during training and test. In red areas the predicted samples are smaller than the original and larger than the original in green areas. Although the predicted signals do not match the original ones perfectly, it is important that the shape of the EMG signals is very well preserved.

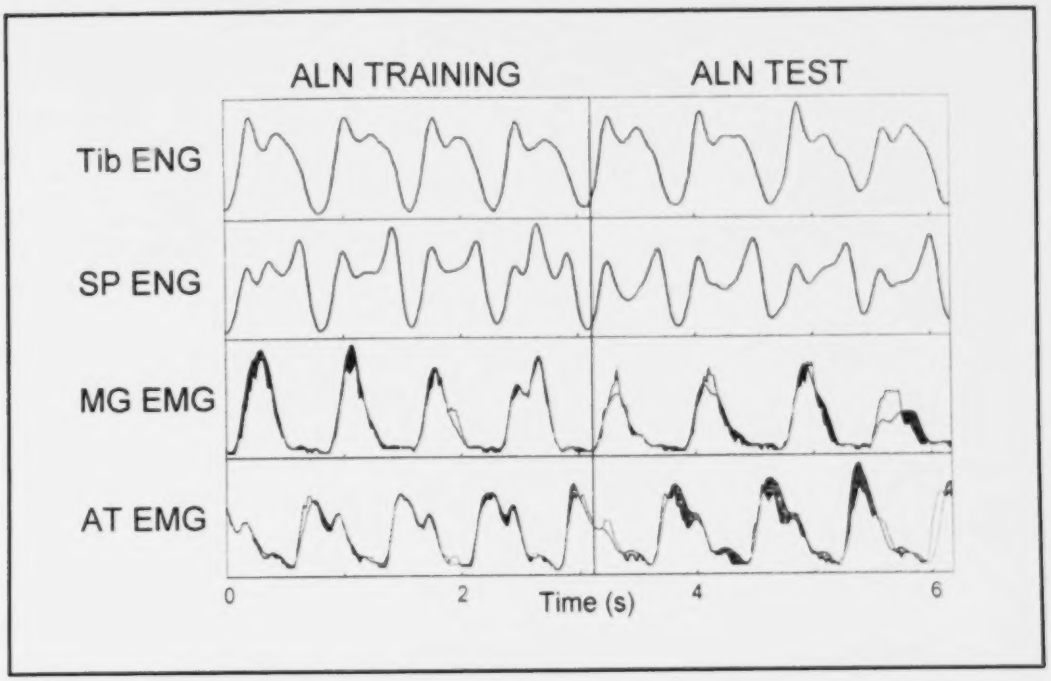

Figure 3.7.17. An example of input and output signals used in ALN training and test. The colored areas in the last two traces show the difference between original and predicted EMGs during training and test.

3.7.2.3 Discussion and conclusion

In this study continuous control of FES-assisted walking was evaluated. The control system is based on the rules generated by adaptive logic networks from neuroelectric and myoelectric signals. Adaptive logic networks were trained to extract invariant characteristics from the input-output relationship, to store these characterictics in the form of artificial neural networks and later to use these networks for production of the control output from the sensory input. Electroneurograms were recorded from the tibial and superficial peroneal nerves and were used to predict the form of electromyograms from medial gastrocnemius and tibialis anterior muscles. Adaptive logic networks were capable of learning the relationship between these inputs and outputs, and of producing the control outputs after successful training when supplied only with sensory inputs. The results achieved are very encouraging. If expressed in variance accounted for

(VAF), the restoration of training control outputs is about 95 %. It is still around 80 % for test signals not used during the training. It is very important that the shapes of predicted EMG signals match the originals very well.

The efficacy of the learning algorithm could improve by increasing the size of the trained ALN trees and the number of voters or by using ALN V.3. To optimize the learning and to obtain satisfactory results in a reasonably short time one has to choose properly the number of past points, the applied preprocessing functions on the input signals and the size of the training set. This is very important because the algorithm in its present form requires a long training. However, its very fast execution of the learned function and easy hardware implementation qualifies it for control applications.

The third step of this experimental design, actual real-time application and miniaturization of this control system, remains to be done in future work.

4. DISCUSSION, CONCLUSIONS AND THE FUTURE WORK

After recognizing the complexity of the rule-design problem for rule-based control systems for FES-assisted walking of subjects with incomplete spinal cord injuries, it was hypothesized that the rules can be automatically generated, using machine learning techniques, from examples recorded in the form of data sets comprising sensory feedback signals and stimulation control signals. The primary target was the upper level controller or the controller at the coordination level in the controller module of the neuroprosthetic device (Figure 1.6.1). To prepare for automatic generation of control rules, manually controlled FES-assisted walking of subjects with incomplete spinal cord injury was studied (Section 3.1). Manually controlled stimulation for walking is important in the rehabilitation of SCI subjects as it provides the subject with a full, easy and reliable way to learn how the muscles would react to different stimulation conditions. It also remains as the backup control system for the stimulation during the development of more sophisticated control systems.

To provide a source of sensory feedback information for automatic control of stimulation, a various sensors were evaluated. It was concluded that an affordable array of force sensors built into the subjects' shoe insoles can provide reliable and reproducible source of feedback information for design of control rules. These sensors are presented in Section 2.3.

The force sensors built into the subject's shoe insoles are used in a simple rule-based control system designed by hand-crafting the control rules (Section 3.2). The development of simple rule-based control of the stimulation not only demonstrated the complexity of the control-system design, but it also introduced automatic control of the stimulation to the subject. This experience was very useful, both for the subject and the researcher during the development of ALN-based

control since the subject acquired an experience of using automatic control and the researcher could learn about practical aspects of the stimulation control for walking. Automatic control of walking usually resulted in faster walking (shorter gait cycle) and longer walking sessions. Although successful, automatic control uncovered a very serious problem of generating control rules for stimulation control, originating in differences between subjects, their stimulation needs, walking skills and patterns. Every subject's walking is different and requires different hand-crafted rules, which limits the transferability of the control rules. Machine learning techniques which can generate the control rules automatically were proposed as a solution for the problem. An Integrated Control System (ICS) based on machine learning technique was designed for fast evaluation of new subjects and generation of rules for control of their stimulation.

Two machine learning techniques for supervised learning were evaluated for automatic generation of control rules by mapping the control output for manual stimulation onto the sensory feedback signals. These techniques are Adaptive Logic Networks (ALNs) and Inductive Learning (IL) which are presented in Section 2.7. IL was already evaluated for similar task of event detection (Kirkwood, 1989; Kirkwood and Andrews, 1989; Heller, 1992), so much more attention was paid to the evaluation of ALNs, which were never before used for detection of time-dependent events.

Evaluation of ALNs for cloning the manual skill of skilled subject or therapist in controlling one channel stimulation for FES-assisted walking is presented in Section 3.3. The capability of ALNs to generate control rules from manually controlled stimulation is demonstrated. In addition, it is demonstrated that the quality of ALN learning depends on the number of sensory feedback channels and that the use of more sensory inputs reduces both training and test errors. To introduce the time-dimension in the learning and prediction process, previous sensory signal samples were used together with the current ones which further reduced the errors. An important feature for control was introduced: an early prediction of stimulation events, which provides a time

interval during which the subject can be informed about the coming stimulation. The ALNs were tested in predicting the stimulation events up to two seconds in advance.

IL was evaluated in a complex environment for cloning the control rules for walking of a subject with a complete spinal cord injury. The results are presented in Section 3.4. It was demonstrated that IL is also capable of cloning the skill of skilled subject in controlling a two-channel stimulation for FES-assisted walking. The design of control rules was done in two steps. In the first step IL was used to measure the relative importance of a particular sensor and, based on this information, to reduce the sensory set from ten sensors to only four of highest importance for particular design. In the second step the reduced sensory set was used as a training set for IL to produce the smallest decision trees with the best generalization. The best results were obtained if previous data samples and differentiated signals were used for training together with the original data samples.

ALN and IL were compared on a larger sample recorded from six subjects. The results of this comparison are presented in Section 3.5. It is demonstrated that, although IL generates its decision trees faster and with lower training error, the ALNs have better generalization. A practical implication of this result is that IL may be better for use in control systems where the training data set represents the domain very well, but it is obvious that the training set acquired during walking of subjects with SCI can not represent all possible situations, because some high-risk situations that would be of interest as a part of training set lead to possible injuries (e.g. instability leading to a fall). Both techniques give better results if previous samples are used together with current ones. Also, both techniques are capable of predicting future stimulation events. Overall performance was better with ALNs than with IL, which was crucial for deciding which machine learning technique to use for the design of integrated control system.

After the evaluation of two different machine learning techniques (MLTs) demonstrated the potential of these techniques for automatic rule-generation for control of FES-assisted walking, the

integrated control system was designed to implement and test the same approach in real-time walking (Section 3.6). The final goal of this experiment was to demonstrate the capability of ALNs to generalize control rules designed by cloning manually controlled walking into automatically controlled walking. To provide additional safety for the subject, restriction rules were designed by hand-crafting, using statistical values determined in the training set, and applied to the control signal produced by ALNs. To test the generalization of the trained ALNs before they were connected to the stimulator as the primary source of stimulation control, several tests were done, first with the control subject and then with SCI subject. Trained ALNs were evaluated on the same data set used for training (ALN training). If their approximation of the output control signal did not contain any functional errors (extra or missing stimuli), they were tested on new data which were not used during the training (Off-line ALN test). If there were no functional errors in predicted output control signals, a similar test was repeated, but this time during real-time manually controlled walking (Walking test). The subject still controlled the stimulation manually, but this time she heard a buzzing noise from the controller interface every time the ALNs predicted the stimulation should be ON. The conditions of this test were the closest possible to those of ALN real-time control of the stimulation. After this test was passed without functional errors, the next was to apply the ALNs in real-time control of stimulation for FES-assisted walking (Walking control). The subject, after standing up from the wheelchair, took one or more manually controlled steps and then the ALN control was switched ON and put in parallel with the manual control, which remained active as a functional override.

The results obtained so far demonstrate the capability of ALNs trained on manually controlled FES-assisted walking to generalize over automatically controlled FES-assisted walking. It was also demonstrated that the generalization is satisfactory not only over the same walking session, but it also extends over the following days. This result implies that this control approach and the system designed may be quite robust and that frequent retrainings of the ALNs (calibration) may not be necessary. It remains to be seen how fast the walking pattern changes requiring new ALN

of time (weeks, months)[8], the ICS can be divided into two parts: FES control fitting station and the FES controller itself. The FES controller can be miniaturized and built into the portable battery operated neuroprosthetic device in two different ways: 1) by transferring it from a desktop computer to a much smaller, but equally or even more powerful notebook computer; 2) by developing custom-designed hardware driven by commercially available microcontroller. Considering the speed of development and the price reduction of high technology today, it is possible to predict that the first of the above options can be done incomparably faster than the other one. The control rules can be generated in the laboratory or at home using an FES control fitting station, which can be based on a small notebook computer with data acquisition extensions. After the control rules are produced, they can be downloaded or transferred in some other way to the FES controller, which makes the FES controller an independent device.

This thesis project has just scratched the surface of the whole new field of implementation of machine learning techniques in the control of biomedical systems. Now that the results of this thesis have proved that the human skill can be automatically transferred into the computer, and that both, the subject and the computer can improve their performance while working together, the next step is to explore the field in which this technique is applicable. Depending on the results of such a study, this technique can be improved and implemented in the form of a commercial device. In the future work the ICS should be applied to stimulation control of as many SCI subjects as possible. More frequent use of the ICS, first in one research center and later in a multi-center trial, would direct its further development into an easy-to-use gait evaluation tool for subjects with incomplete spinal cord injuries. It can provide the fastest way to design the control rules for automatically controlled FES-assisted walking and to demonstrate to the subject what

---

[8] There are some encouraging preliminary results which demonstrate good generalization even four months after the training.

automatically controlled walking is all about. With the ICS all this can be achieved during the same walking session.

Although ALNs have very low generalization error, the restriction rules should be present in future developments based on this approach either in the form of a functional filter after the ALNs or in the form of *a priori* knowledge built into the ALN trees. The latter is already possible with ALN V.3.

Even before the ALN's capability to generalize from manually to automatically controlled walking was demonstrated, the evaluation of ALNs for continuous output control was done. This type of control falls under lower level control or control at the actuator level in the controller module of the neuroprosthetic device (Figure 1.6.1). The biggest challenge was to try to use artificial neural networks to clone the function of the natural reflexes and control loops. Since this experiment required implantation of recording electrodes in living subject, the experiment was done in chronic cats. Electroneurograms (ENGs), recorded using cuff electrodes, implanted around the tibial and superficial peroneal nerves, were used as the sensory feedback information, and electromyograms (EMGs), recorded using epimysial electrodes, implanted on the tibialis anterior and medial gastrocnemius muscles, were used as the control output signals.

In the first experiment, ALNs were used to approximate the transfer function between continuous ENG inputs and simplified (binary) EMG outputs. This was similar to experiments described previously with ON/OFF type control. The ALNs performed comparably to a simple hand-crafted rule-based control system. The advantage of the ALNs was that they could predict not only the existence of control pulses, but also their duration (Section 3.7.1). This work also demonstrated that natural sensory signals can be used as sensory feedback information in control systems for locomotion.

In the second experiment, the ALNs were used with the same input and output signal sources, but this time for approximation of the transfer function between continuous ENG inputs and continuous EMG outputs. Optimization of the ALN V.2 training was done for this particular design.

Results demonstrated ALN's capability of approximating very complex transfer functions and predicting the full shape of rectified and integrated EMGs from ENGs. In future work the predicted EMGs can be used to modulate stimulation pulse trains and delivered to the appropriate muscles, whose motoneurons are chemically blocked, to produce automatically controlled FES-assisted walking.

The basic concept of this technique can be improved by implementing reinforced learning instead (or in parallel) with supervised learning technique, which will provide this control system with self-adaptation feature. This will provide continuous learning during the use of the control system.

5. BIBLIOGRAPHY

Aleksander I. (1970), Some phychological properties of digital learning nets. International Journal of Man-Machine Studies 2, pp. 189-212.

Andrews B.J., and Bajd T., (1984), Hybrid Orthoses for paraplegics, in *Proc. (suppl) 8th Internat. Symp. External Control of Human Extremities*, Dubrovnik, pp. 55-57.

Andrews B.J., and Baxendale R.H., (1986), A hybrid orthosis incorporating artificial reflexes for spinal cord damaged patients, *Journal of Physiology*, 380, 19P.

Andrews B.J., Baxendale R.H., Barnett R., Phillips G.F., Paul J.P., and Freeman P.A., (1987), A Hybrid Orthosis for Paraplegics Incorporating Feedback Control, *in Advances in External Control of Human Extremities IX*, Edited by Popovic D., pp. 297-311.

Andrews B.J., Baxendale R.H., Barnett R., Phillips G.F., Yamazaki T., Paul J.P. and Freeman P.A., (1988), Hybrid FES orthosis incorporating closed loop control and sensory feedback, *J. Biomed. Eng.*, Vol. 10, pp. 189-195.

Andrews B.J., Barnett R.W., Phillips G.F., Kirkwood C.A., Donaldson N., Rushton D.N., and Perkins T.A., (1989), Rule-Based control of a hybrid FES orthosis for assisting paraplegic locomotion, *Automedica*, Vol. 11, pp. 175-200.

Armstrong W.W., (1971), The trainable digital apparatus, *U.S. Pat. No. 3,613,084*, issued Oct. 12, 1971.

Armstrong W.W., and Bochmann G.v., (1972), A convergence theorem for logical network applications, *Publ. No. 95*, Département d'Informatique, Université de Montréal.

Armstrong W.W., and Godbout G., (1974), Use of Boolean Tree Functions to Perform High-speed Pattern Classification and Related Tasks, *Publ. No. 53*, Département d'Informatique, Université de Montréal.

Armstrong W.W., (1976), Adaptive Boolean Logic Element, *U.S. Patent 3,934,231*, Jan. 20, 1976, assigned to Dendronic Decisions Limited.

Armstrong W.W., and Gecsei J. (1979), Adaptation Algorithms for Binary Tree Networks, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-9, No. 5, pp. 276-285.

Armstrong W.W., Liang J.D., Lin D., and Reynolds S., (1990), Experiments using Parsimonious Adaptive Logic, *Technical Report TR 90-30*, Department of Computing Science, University of Alberta, Edmonton.

Armstrong W.W., Dwelly A., Liang J.D., Lin D., and Reynolds S., (1991), Learning and generalization in adaptive logic networks, in *Proc. ICANN '91*, Elsevier Press, Amsterdam, pp. 1173-1176.

Armstrong W.W. and Thomas M.M. (1994), *Dendronic Decisions Atree 3.0 beta release 1*, ALN Theory, A Practical Guide to Approximating Relations.

Axon Instruments Inc., (1989), *Axotape 1.2 Users Manual*

Barr R.E., and Chan E.K.Y. (1986), Design and implementation of digital filters for biomedical signal processing, *J. Electophy.Tech.*, Vol. 13, pp. 73-93, 1986.

Beckmann J., Daunicht W.J., and Hömberg V., (1992), Control of paraplegic patient model by neuroprosthetic networks, in *Proc. of ICANN-92*, ed. Aleksander I. and Taylor J., Brighton, UK.

Beebe D.J., Hsieh A.S., and Denton D.D., (1994), A finger-mounted silicon tactile sensor, in *Proc 16th Annual International IEEE - EMBS Conference*, Baltimore, USA, pp. 834-835

Bélanger M., (1992), *SELPOS & ENSAVG: Operating Manuals*, Division of Neuroscience, University of Alberta.

Benes V., (1965), *Spinal Cord Injury*, Publ. Bailliere, Tindall & Cassell, London, U.K.

Benton L.A., Baker L.L., Bowman B.R. and Waters R.L. (1981), *Functional Electrical stimulation - A Practical Clinical Guide, 2nd ed.*, Rancho Los Amigos, Rehabilitation Engineering Center, Rancho Los Amigos Hospital, Downey, California, pp. 1-10.

Blahut R.E. (1987), *Principles and Practice of Information Theory*, Addison-Wesley Publishing Co., Reading, Massachusetts.

Bochmann G.v., and Armstrong W.W., (1974), Properties of Boolean Functions with a tree Decomposition, *BIT 14*, pp. 1-13.

Braune W., and Fisher O., (1987), On the Movement of the Foot and the Forces Acting on It, in *The Human Gait, Chapter 4*, Springer-Verlag, pp. 255-313.

Cayaffa J.J., (1981), Anatomy and vascular supply of the spinal cord, in *Radiology of Spinal Cord Injury*, edited by Calenoff L., Publ. The C.V. Mosby Company.

Chao E.Y.S., and Cahalan T.D., (1990), Kinematics and Kinetics of Normal Gait, in *Clinics in Physical Therapy: Gait in Rehabilitation*, Edited by Smidt G.L., pp. 45-63.

Chizeck H.J., (1992), Adaptive and nonlinear control methods for neuroprostheses, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 298-327.

Crago P.E., Chizeck H.J., Neuman N., and Hambrecht F.T., (1986), Sensors for use with functional neuromuscular stimulation. *IEEE Trans.Biomed.Eng.*, Vol.33, pp.256-268.

Cull J.G. and Hard·· R.E., (1977), Physical Medicine and Rehabilitation Approaches, in *Spinal Cord Injury*, Publ. Charles C. Thomas, Springfield, Illinois, U.S.A.

Curtis H.A. (1962), *A New Approach to the Design of Switching Circuits*, Princeton, NJ: van Nostrand.

Durfee. W.K., and Hausdorff J.M., (1990), Regulating knee joint position by combining electrical stimulation with a controllable friction brake, *Ann. Biomed. Eng.*, Vol. 18, pp. 575-596.

Durfee W.K., (1992)., Model Identification in Neural Prosthesis Systems, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 58-87.

Ferencz D.C., Jin Z, and Chizeck H.J., (1993), Estimation of Center-of-Pressure During Gait Using an Instrumented Ankle-Foot Orthosis, in *Proc 15th Annual International IEEE - EMBS Conference*, San Diego, U.S.A., Vol. 2, pp. 981-982.

Ferrigno G., and Pedotti A., (1985), ELITE: A Digital Dedicated Hardware System for Movement Analysis Via Real-Time TV Signal Processing, *IEEE Trans.Biomed.Eng.*, Vol. BME-32, No. 11, pp. 943-950.

Frankel H.L., Hancock D.O., Hyslop G., Melzak J., Michaelis L.S., Ungar G.H., Vernon J.D.S., and Walsh J.J., (1969), The Value of Postural Reduction in the Initial Management of Closed Injuries of the Spine with Paraplegia and Tetraplegia, *Paraplegia*, Vol. 7, pp. 179-192.

Franken M. Henry, (1994), Control System Design for Walking Neuroprostheses, Ph. D. Thesis, University of Twente, Enschede, the Netherlands.

Geddes L.A., (1994), The First Stimulators, *IEEE Engineering in Medicine and Biology Magazine*, Vol. 13, No. 4, pp. 532-542.

Graupe D., and Cline W.K., (1975), Functional Separation of EMG Signals via ARMA Identification Methods for Prosthesis Control Purposes, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-5, pp. 252-259.

Graupe D., Kohn K.H., Kralj A., and Basseas S., (1983), Patient Controlled Electrical Stimulation via EMG Signature Discrimination for Providing Certain Paraplegics with Primitive Walking Functions, *J. Biomed. Eng.*, Vol. 5, pp. 220-226.

Graupe D., Kohn K.H., and Basseas S., (1988), Above- and below-lesion EMG pattern mapping for controlling electrical stimulation of paraplegics to facilitate unbraced walker-assisted walking, *J. Biomed. Eng.*, Vol. 10, pp. 305-311.

Graupe D., (1989), EMG Pattern Analysis for Patient-Responsive Control of FES in Paraplegics for Walker-Supported Walking, *IEEE Trans.Biomed.Eng.*, Vol. 36, NO. 7, pp. 711-719.

Grillner S., and Zangger P., (1985), How detailed is the central pattern generation for locomotion?, *Brain Research*, Vol. 88, pp. 367-371.

Guttmann L., (1976), *Spinal Cord Injuries: Comprehensive Management and Research*, Publ. Blackwell Scientific Publications, 2nd edition.

Halpern P.H. (1966), Generalized Self Synthesizer, *U.S. Pat. No. 3,262,101*, issued July 19.

Hambrecht F.T., (1992), A Brief History of Neural Prostheses for Motor Control of Pralyzed Extremities, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 3-14.

Handa Y., Naito A., Ichie M., Handa T., Matsushita N., and Hoshimiya N., (1987), EMG-based stimulation patterns of FES for the paralyzed upper extremities, in *Advances in External Control of Human Extremities IX*, Edited by Popovic D., pp. 329-337.

Haugland M.K., Hoffer A., and Sinkjaer T., (1994), Skin Contact Force Information in Sensory Nerve Signals Recorded by Implanted Cuff Electrodes, *IEEE Trans.Rehab.Eng.*, Vol. 2, No. 1, pp. 18-28.

Haugland M.K. and Hoffer A., (1994a), Slip Information Provided by Nerve Cuff Signals: Application in Closed-Loop Control of Functional Electrical Stimulation, *IEEE Trans.Rehab.Eng.*, Vol. 2, NO. 1, pp. 29-36.

Haugland M.K. and Hoffer A., (1994b), Artifact-Free Sensory Nerve Signals Obtained from Cuff Electrodes During Functional Electrical Stimulation of Nearby Muscles, *IEEE Trans.Rehab.Eng.*, Vol. 2, NO. 1, pp. 37-40.

Heller B.W., (1992), *The Production and Control of Functional Electrical Stimulation Swing-Through Gait*, Ph.D. Thesis, Bioengineering Unit, University of Strathclyde, UK.

Heller B.W., Rijkhoff N.J.M., Veltink P.H., Rutten W.C.L., and Andrews B.J., (1993), Reconstructing muscle activation during normal walking: a comparison of symbolic and connectionnist machine learning techniques, *Biol. Cybern.*, Vol. 69., pp. 327-335.

Hodgkin A.L. and Huxley A.F., (1945), Resting and action potentials in single nerve fibres, *Journal of Physiology*, Vol. 104, pp. 176-195.

Hodgkin A.L. and Huxley A.F., (1952), A quantitative description of membrane current and its application to conduction and excitation in nerve, *Journal of Physiology*, Vol. 107, pp. 500-544.

Hoffer J.A., Haugland M., and Li T., (1989), Obtaining Skin Contact Force Information from Implanted Nerve Cuff Recording Electrodes, in *Proc 11th Annual International IEEE - EMBS Conference*, pp. 928-929.

Hoffer J.A., and Haugland M.K., (1992), Signals from tactile sensors in glabrous skin suitable for restoring motor functions in paralyzed humans, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 99-125.

Hollerbach J.M., and Bennet D.J., (1992), Feed-forward versus Feeback Control of Limb Movements, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 129-147.

Hoshimiya N., Takahashi M., Handa Y., and Sato G., (1976), Basic studies on electrophrenic respiration (Pt. 1): Electrophrenic respirator synchronized with phrenic nerve impulses, *Med.Biol.Eng.*, Vol. 14, pp. 387-394.

Hruska S.I., Kuncicky D.C., and Lacher R.C., (1991), Hybrid Learning in Expert Networks, in *Proc. of IJCNN-91*, Seatle, WA, Vol. II, pp. 117-120.

Interlink Electronics, (1990a), Force and Position Sensing Resistors: An Emerging Technology, *Technical Overview*, Rev: 2/90.

Interlink Electronics, (1990b), *TechNotes*, New Release, September 1990.

Jackson R.W., (1981), *The Care and Management of Spinal Cord Injuries*, by Bedbrook G.M., Publ. Springer-Verlag.

Jaeger R.J., (1986), Design and Simulation of Closed-Loop Electrical Stimulation Orthoses for Restoration of Quiet Standing in Paraplegia, *J. Biomechanics*, Vol. 19, No. 10, pp. 825-835.

Kabat H., (1954), Studies on neuromuscular dysfunction: XV. role of central facilitation in restoration of motor function in paralysis, *Arch.Phys.Med.Rehabil.*, Vol. 33, pp. 521-533.

Kahn J., (1994), *Principles and practice of electrotherapy*, Churchill Livingstone Inc. New York, pp. 75-106

Kandel E.R., Schwartz J.H., and Jessel T.M., (1991), *Principles of Neural Science*, 3rd Edition, Elsevier, New York

Karsai G., (1991), Learning to Control: Some Practical Experiments with Neural Networks, in *Proc. of IJCNN-91*, Seatle, WA, Vol. II, pp. 701-707.

Kelly M.F., Parker P.A., and Scott R.N., (1990), The Application of Neural Networks to Myoelectric Signal Analysis: A Preliminary Study, *IEEE Trans.Biomed.Eng.*, Vol. 37, No. 3, pp. 221-230.

Kirkwood C.A., and Andrews B.J., (1988), Rule-based control for FES implemented using firmware transitional logic, in *Proc 10th Annual International IEEE - EMBS Conference*, New Orleans, USA.

Kirkwood C.A., (1989), *Evaluation of inductive learning techniques applied to gait event detection for rule based control of F.E.S.*, Ph.D. thesis, Bioengineering Unit, University of Strathclyde, Glasgow, UK.

Kirkwood C.A., and Andrews B.J., (1989), Finite-state control of FES systems: Application of AI inductive learning techniques, in *Proc. 11th Annual International IEEE - EMBS Conference*, Seatle, USA, pp. 1020-1021.

Kobetic R., (1994), Advancing step by step, *IEEE Spectrum*, October 1994, pp. 27-31.

Kobetic R., and Marsolais E.B., (1994), Synthesis of Paraplegic Gait with Multichannel Functional Neuromuscular Stimulation, *IEEE Trans.Rehab.Eng.*, Vol. 2, No. 2, pp. 66-79.

Kostov A., Stein R.B., Armstrong W.W., and Thomas M., (1992), Evaluation of Adaptive Logic Networks for Control of Walking in Paralyzed Patients", in *Proc. 14th Annual IEEE - EMBS Conference*, Paris, France, vol. 4, pp. 1332-1334.

Kostov A., Popovic D., Stein R.B., and Armstrong W.W., (1993), Learning of EMG-patterns by Adaptive Logic Networks, in *Proc 15th Annual International IEEE - EMBS Conference*, San Diego, California, U.S.A., Vol. 3, pp. 1135-1136.

Kostov, A., Stein, R.B., Popovic, D., and Armstrong, W.W. (1994), Improved Methods for Control of FES, in *Proc. of IFAC Symposium on Modeling and Control in Biomedical Systems*, Galveston, Texas, March 27-30, 1994, pp. 422-427.

Kostov, A., Andrews, B.J., Popovic, D.B., Stein, R.B., and Armstrong, W.W. (1995a), Machine Learning in Control of Functional Electrical Stimulation Systems for Locomotion, *IEEE Trans.Biomed.Eng., (accepted for publication in June 1995).*

Kostov, A., Andrews, B.J., and Stein R.B., (1995b), Inductive Machine Learning in Control of FES-assisted Gait After Apinal Cord Injury, the *5th Vienna International Workshop on Functional Electrical Stimulation*, Aug. 17-19, 1995, *(accepted)*

Kostov, A., Andrews, B.J., Stein, R.B., and Popovic D.B. (1995c), The use of inductive machine learning to select the most appropriate sensors to predict events: Application in automatic control of FES-assisted gait in paraplegia, *(paper in preparation)*

Kostov, A., Stein, R.B., Popovic, D.B., and Armstrong, W.W. (1995d), Automatic generation of FES-control rules: Application to real-time control of locomotion for subjects with incomplete spinal cord injury, *(paper in preparation)*

Kralj A.R., and Bajd T., (1989), *Functional Electrical Stimulation: Standing and Walking after Spinal Cord Injury*, CRC Press, Inc., Boca Raton, Florida

Lan N., Crago P.E. and Chizeck H.J., (1991), Feedback control methods for task regulation by electrical stimulation of muscles, *IEEE Trans.Biomed.Eng.*, Vol. 38, pp. 1213-1223.

Lazorthes Y., Morucci J.-P., and Clemente G., (1985), Biotechnological Basis of Neurostimulation, Chapter 2 in *Neurostimulation: An Overview*, eds: Lazorthes Y. and Upton A.R.M., Futura Publishing Company, Inc., Mt. Kisco, New York, pp. 11-41

Lee K.H. and Johnston R. (1976), Electrically induced flexion reflex in gait training of hemiplegic patients: induction of the reflex, *Arch.Phys.Med.Rehabil.*, Vol. 57, pp. 311-314.

Lee R.J. (1967), The networks of Artrons, *U.S. Pat. No. 3,327,291*, issued June 20.

Liberson W.T., Holmquest H.J., Scott D. and Dow M., (1961), Functional electrotherapy, stimulation of the peroneal nerve synchronized with the swing phase of the gait of hemiplegic patients, *Arch.Phys.Med.Rehabil.*, Vol. 42, pp. 101-105.

Loeb G.E., Zamin C.J., Schulman J.H., and Troyk P.R., (1991), Injectable microstimulator for functional electrical stimulation, *Med. & Biol. Eng. & Comput.*, Vol. 29, pp. NS13-NS19.

Loparo K., and Teixeira E., (1990), A new approach for adaptive control of a nonlinear system using neural networks, *IEEE Intl. Conf. Systems, Man and Cybernetics*, pp. 38-40.

Lord M. (1981), Foot Pressure Measurement: A Review of Methodology, *Journal of Biomedical Engineering*, Vol. 3, pp. 91-99.

Lord M., Reynolds D.P. and Hughes J.R. (1986), Foot Pressure Measurement: A Review of Clinical Findings, *Journal of Biomedical Engineering*, Vol. 8, pp. 283-293.

Marsolais E.B., Ko W., and Masiello A., (1985), Finger switch for a portable microprocessor system to restore walking in paraplegics, in *Proc. 8th Annual RESNA Conf.*, pp. 376-378.

Marsolais E.B., and Edwards B.G., (1988), Energy cost of walking and standing with functional neuromuscular stimulation and long leg braces, *Arch.Phys.Med.Rehab.*, Vol 69, pp. 243-249.

McClelland M., Andrews B.J., Patrick J., and Masrai E., (1987), FES Augmentation of the Oswestry Parawalker Orthosis, *Paraplegia*, Vol. 25, pp. 32-38.

Michie D. and Chambers R.A. (1968), Boxes: an experiment in adaptive control. In Dale E. and Michie D. eds: Machine Intelligence 2, Edinburgh University Press, Edinburgh, pp. 137-152

Minsky M. (1961), Steps toward artificial intelligence, *in Proc. IRE*, Vol. 49, pp. 8-30, 1961. Reprinted in *Computers and Thought*, E.Feigenbaum and J. Feldman, Eds. New York: McGraw-Hill, 1963, pp. 406-450.

Minsky M. and Papert S. (1969), *Perceptrons: An Introduction to computational geometry*, M.I.T. Press.

Mulder A.j., Verheyen J.M.E., and Nijmeijer H, (1987), Closed-Loop Control of Stimulation during Standing, in *Proc. of Advances in External Control of Human Extremities IX*, edited by Popovic D , Dubrovnik, Yugoslavia, pp. 275-282.

National Instruments, (1993), *IEEE 488 and VXIbus Control, Data Acquisition, and Analysis*, Software and Hardware Product Catalogue, pp. 1-5 to 1-14 and 3-28 to 3-33.

Nikolic Z. and Popovic D., (1992), A low-noise gaited amplifier for closed-loop control FES systems, in *Proc 14th Annual International IEEE - EMBS Conference*, Paris, pp. 1644-1645.

Nikolic Z. and Popovic D., Stein R.B., and Kenwell Z., (1994), Instrumentation for ENG and EMG recordings in FES Systems, *IEEE Trans.Biomed.Eng.*, Vol. 41, No. 7, pp. 703-706.

Oliver M., Zarb G., Silver J., Moore M., and Salisbury V., (1988), *Walking Into Darkness: The Experience of Spinal Cord Injury*, Publ. Macmillan Press, Ltd., London, U.K.

Osterholm J.L., (1978), *The Pathophysiology of Spinal Cord Trauma*, Publ. Charles C. Thomas, Springfield, Illinois, U.S.A.

Ozer M.N., (1988), *The management of Persons with Spinal Cord Injury*, Publ. Demos Publications, New York.

Payne P.A., (1989), Transducers, sensors, and instrumentation in clinical biomechanics, *J. Biomed. Eng.*, Vol. 11, pp. 180-184.

Peckham P.H., and Keith M.W., (1992), Motor prostheses for Restoration of Upper Extremity Function, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 162-187.

Penny and Giles Biometrics Limited, (1994), *Goniometer and Torsiometer Operating Manual*.

Perry J. (1992), *Gait Analysis: Normal and Pathological Function*, Publ. SLACK Inc.

Petrofsky J.S., Phillips C.A., and Stafford D.E., (1984), Closed-Loop Control for Restoration of Movement in Paralyzed Muscle, *Ortopedics*, Vol. 7, pp. 1289-1302.

Popovic D., Tomovic R., and Schwirtlich L., (1989), Hybrid Assistive System - The Motor Neuroprosthesis, *IEEE Trans.Biomed.Eng.*, Vol. 36, pp. 729-737.

Popovic D., Tomovic R., Tepavac D., and Schwirtlich L., (1991), Control aspects of an active above-knee prosthesis, *International Journal on Man-Machine Studies*, 1991, Academic Press, London, Vol. 35, pp. 751-767.

Popovic D., and Raspopovic V., (1992), Afferent signals in palmar digital nerves, in *Proc. Fourth Vienna Int. Workshop on FES*, 1992, pp. 105-108.

Popovic D., Stein R.B., Jovanovic K., Dai R.C., Kostov A., and Armstrong W.W., (1993), Sensory Nerve Recording for Closed-Loop Control to Restore Motor Functions, *IEEE Trans.Biomed.Eng.*, Vol. 40, No. 10, pp. 1024-1031.

Prochazka A. (1993), Comparison of Natural and Artificial Control of Movement, *IEEE Trans.Rehab.Eng.*, Vol. 1, No. 1, pp. 7-17.

Reswick J.B. (1973), A Brief History of Functional Electrical Stimulation, in *Neural Organization and Its Relevance to Prosthetics*, eds. W.S. Fields and L.A. Leavitt, New York: Intercontinental Medical Book Corp., 1973, pp. 3-13.

Rieser T.V., Mudiyam R., and Waters R.L., (1985), Orthopedic Evaluation of Spinal Cord Injury and Management of Vertebral Fractures, in *Spinal Cord Injury*, ed. Adkins H.V., Publ. Churchill Livingstone.

Rothwell J.C., (1987), *Control of Human Voluntary Movement*, Aspen Publishers, Inc., Rockville, Maryland

Schley C., Chauvin Y., and Mittal-Henkle V., (1991), Integrating Optimal control with Rules using Neural Networks, in *Proc. of IJCNN-91*, Seatle, WA, Vol. II, pp. 759-763.

Schwirtlich L, and Popovic D., (1984), Hybrid orthoses for deficient locomotion, in *Proc 8th Internat. Symp. External Control of Human Extremities*, Dubrovnik, pp. 23-32.

Sethi I.K. and Sarvarayudu G.P.R., (1982), Hierarchical classifier design using mutual information, *IEEE Trans.Pattern.Anal.Mach.Intell.*, Vol. 4, pp. 441-445.

Sinkjær T., Hinge B., Jørgansen A., Jensen M.L., and Haugland M.K., (1992), Whole Sensory Nerve Recordings With Spiral Nerve Cuff Electrode, in *Proc 14th Annual International IEEE - EMBS Conference*, Paris, 1992, vol. 4, pp. 1330-1331.

Smith D., and Stanford P., (1990), A Random Walk in Hamming Space, in *Proc. Int. Joint Conf. on Neural Networks*, San Diego, vol. 2, pp. 465-470.

Solomonow M., Baratta R., Shoji H., and D'Ambrosia R.D., (1988), The myoelectric signal of electrically stimulated muscle during recruitment: An inherent feed-back parameter for a closed-loop control scheme, *IEEE Trans.Biomed.Eng.*, Vol. BME-33, NO. 8, pp. 735-745.

Solomonow M., (1992), Biomechanics and Physiology of a Practical Functional Neuromuscular Stimulation Powered Walking Orthosis for Paraplegics, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 202-232.

Stein R.B., Charles D., Davis L., Jhamandas J., Mannard A., Nichols T.R., (1975), Principles underlying new methods for chronic neural recording, *J. Neurol. Sci.*, Vol 2, pp. 235-244.

Stein R.B., Nichols T.R., Jhamandas J., Davis L., and Charles D., (1977), Stable long-term recordings from cat paripheral nerves, *Brain Research*, Vol. 128, pp. 21-38.

Stein R.B., Charles D., Gordon T., Hoffer J.A. and Jhamandas J., (1978), Impedance properties of metal electrodes for chronic recording from mammalian nerves, *IEEE Trans.Biomed.Eng.*, Vol. BME-25, pp. 532-537.

Stein R.B., Kostov A., Belanger M., Armstrong W.W., and Popovic D., (1992), Methods to control functional electrical sitmulation, in *Proc. Ist FES Intern. Symp on FES*, Sendai, July 22-26, Japan.

Stein R.B., Bélanger M., Wheeler G., Wieler M., Popovic D., Prochazka A., and Davis L.A., (1993), Electrical Systems for Improving Locomotion After Incomplete Spinal Cord Injury: An Assesment, *Arch Phys Med Rahabil.*, Vol 74, pp. 954-959.

Stanic U., and Trnkoczy A., (1974), Closed-Loop Positioning of Hemiplegic Patient's Joint by means of Functional Electrical stimulation, *IEEE Trans.Biomed.Eng.*, Vol. BME-21, No. 5, 365.

Tashman S., and Zajac F.E., (1992), Control of Multijoint Lower Limb Motor Tasks with Functional Neuromuscular Stimulation, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 252-278.

Tepavac D., Popovic M., Kostov A., Nikolic Z., and Popovic D., (1992), A portable recording system for assesment of functional electrical stimulation assisted locomotion, in *Proceedings of the 1st International FES Symposium*, Sendai, Japan, pp. 130-134.

Tomovic R., and McGhee R.B., (1966), A Finite State Approach to the Synthesis of Bioengineering Control Systems, *IEEE Trans. on Human factors in Electronics*, Vol-HFE 7, No 2, pp. 65-69.

Tomovic R., Vukobratovic M., and Vodovnik L., (1972), Hybrid actuator for orthotic systems: Hybrid assistive systems, in *Proc. 4th ECHE.*, Dubrovnik, pp. 73-79.

Tomovic R., (1984), Control of assistive systems by external reflex arcs, in *Advances in External Control of Human Extremities VIII*, Yugoslav Commitee for ETAN, Belgrade, pp. 7-21.

Tomovic R., Popovic D., and Tepavac D., (1987), Adaptive reflex control of assistive systems, in *Advances in External Control of Human Extremities IX*, Yugoslav Commitee for ETAN, Belgrade, pp. 207-214.

Tomovic R., (1989), Transfer of Motor Skills to the Machine, *Int. J. of Robotics and Computer-Integrated Manufacturing*, Vol. 5, No. 2/3.

Van Doren C.L., and Riso R.R., (1991), Synthetic sensory feedback for motor prostheses, *J.Neurol.Rehabil.*

Veltink P.H., Rijkhoff N.J.M., and Rutten W.L.C., (1990), Neural Networks for Reconstructing Muscle Activation from External Sensor Signals During Human Walking, in *Proc. of IEEE Intelligent Motion Control*, Istanbul, pp. 469-473.

Viewpoint Software Solutions, Inc., (1993), *Downshift for LabVIEW for Windows.*

Webster J.G. (1992), Artificial sensors suitable for closed-loop control of FNS, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 88-98.

Wertsch J.J., Webster J.G., Tompkins W.J., (1992), A portable insole plantar pressure measurement system, *Journal of Rehabilitation Research and Development*, Vol. 29, No. 1, pp. 13-18.

Whittle M.W. (1991), *Gait Analysis: An introduction*, Butterworth Heinemann.

Wilhere G.F., Crago P.E. and Chizeck H.J., (1985), Design and Evaluation of a Digital Closed-Loop Controller for the Regulation of Muscle Force by Recruitment Modulation, *IEEE Trans.Biomed.Eng.*, Vol. BME-32, No. 9, pp. 668-676.

Winter D.A., Sidwall H.G. and Hobson D.A. (1974), Measurement and reduction of noise in kinematics of locomotion, *Journal of Biomechanics*, Vol. 7, pp. 157-159.

Winter D.A. (1990), *Biomecahnics and Motor Control of Human Movement*, Second Edition, John Wiley&Sons, Inc.

Winter D.A., (1991), *The Biomechanics and Motor Control of Human Gait: Normal, Elderly and Pathological*, Second Edition, University of Waterloo Press.

Winters J.M., and Woo S.L., eds., (1990), *Multiple Muscle Systems*, New York: Springer-Verlag.

Yamaguchi G.T., Sawa A., Moran D., Fessler M., and Winters J, (1990), A survey of human musculotendon actuator parameters, in *Multiple Muscle Systems*, eds. Winters, J.M., and Woo, S.L., New York: Springer-Verlag.

Yamaguchi G.T., and Zajac F.E., (1990), Restoring unassisted gait to paraplegics via functional electrical stimulation: A computer simulation study, *IEEE Trans.Biomed.Eng.*, Vol. 37, 886-902.

Zahalak G.I., (1992), An Overview of Muscle Modeling, in *Neural Prostheses: Replacing motor function after disease or disability*, eds. R.B. Stein, H.P. Peckham and D.B. Popovic, New York: Oxford Univ. Press, pp. 17-57.

Zhu H., Maalej N., Webster J.G., Tompkins W.J., Bach-Y-Rita P., and Wertsch J., (1990), An Umbilical Data-Acquisition System for Measuring Pressures Between the Foot and Shoe, *IEEE Trans.Biomed.Eng.*, Vol. 37, No. 9, pp. 908-911.

Appendix A : DATA ACQUISITION SYSTEMS

AXOTAPE

The Axotape system enables a microcomputer to continuously acquire and display multichannel data. Functionally, it is similar to a multichannel tape recorder, but it takes advantage of the recent dramatic advances in mass storage devices and records directly to any logical storage drive on a computer. There are numerous advantages of the Axotape system over a conventional tape recorder (Axon Instruments, 1989). Data can be recorded directly to the disk with simultaneous screen display for easy monitoring. In order to monitor the activity and range of the input signal, data can also be displayed on the screen without recording it. In this mode Axotape is working like an oscilloscope. For maximum-frequency recording conditions, data can be recorded to the disk without any display. The 12-bit recording resolution provides a noise level of only ±0.02%, which, for a ±10V signal is ±5mV, while a conventional FM tape recorder has a 45 dB signal to noise ratio corresponding to about 50 mV of noise for the same recording range. After data are collected, they are already in the correct format and readily accessible for further computer analysis. In addition, the Axotape program allows one to move quickly to any specified time in the data file. For recordings of long duration this provides an enormous saving of time.

After data have been recorded, they can be played back for review using a range of timebase and display options. Measurements can be made using a pair of cursors and stored in ASCII file. Selected portions of the data can be plotted to a high-resolution graphics device or written to an ASCII file so that it can be read by other programs.

The Axotape system consists of the following hardware and software components:

A/D data acquisition board - Axon TL-1-125 is an extension board which plugs into one of the ISA extension slots of an IBM PC or compatible computer. It can sample up to sixteen channels of data with the maximal sampling rate of 125 kHz when recording only one channel. The input range is adjustable up to ±10V. The board converts continuous signals in binary numbers with the quantization resolution of 12 bits. The data are stored in a proprietary FETCHEX 5.2 binary data file format (Axon Instruments, 1989), which also contains all recording parameters stored in the file header. The information stored in the data file header provides a researcher with the opportunity to write a customized program for direct retrieval and analysis of the stored data.

Axotape program - provides the following functions: Record, DisplayOnly, AcquireOnly, Stop, Playback, Configure, Help, and Quit. The function names are self-explanatory and the details on all of them can be found in the program's documentation. Besides cursor measurements in the Playback function, the program does not have any other data analysis functions or tools. The proprietary FETCHEX 5.2 data file format is an original one and it is not recognized by any commercial data analysis programs. Thus, to access recorded data and make it available to the commercial data analysis programs, a conversion program HEX_ASC.EXE (see Appendix C) was written in the C computer language, which reads the Axotape data files and converts them into standard ASCII format. A variation of the same program was used to prepare the data files for machine learning programs by modifying a new ASCII header to accommodate the requirements of these programs.

AT-MIO-16DH DATA ACQUISITION BOARD AND LABVIEW FOR WINDOWS

The second input/output system, which was used as a platform on which the real-time integrated control system (ICS) was built, also consists of hardware and software parts. The hardware is in form of an AT-MIO-16DH multifunctional I/O extension board (National Instruments, 1993), which

plugs into one of the ISA extension slots of an IBM PC or compatible computer. It can do 12-bit analog to digital conversion of data with a maximal sampling rate of 110 kHz per channel. The board can be configured to record from up to sixteen single-ended channels or eight differential channels. The input range is jumper-selectable to ±10 V, ±5 V or 0 to 10 V. The gain can be programmatically adjusted to 1, 2, 4, or 8. The board has two 12-bit analog outputs, jumper-configurable to 0 to 10 V unipolar mode or ±10 V bipolar mode. The current drive capability is ±2 mA. It also has 32 TTL-compatible digital lines and four timing I/O channels (3 counter/timers and one frequency output).

Basic software support for the board comes in the form of NI-DAQ, which is a library of functions for controlling National Instruments PC-based data acquisition boards. The function library is provided for DOS and for Windows. DOS functions can be called from programs developed in Microsoft C, QuickBasic, Turbo C and Turbo Pascal. Windows NI-DAQ is in the form of a Microsoft Windows 3.x DLL (dynamic-link library) callable from any Windows application developed in Microsoft C with SDK (software development kit), Microsoft Visual C++, Microsoft Visual Basic, Turbo Pascal for Windows and Borland C++.

However, National Instruments also provides higher level software support for its data acquisition boards. Two integrated development environments are currently available: LabWindows (for DOS and Windows) and LabVIEW (LV) for Windows. The first one, LabWindows enhances Microsoft QuickBasic and C with an interactive development environment, containing function panels to generate source code, and libraries for data acquisition, instrument control, data analysis and presentation. The second one, LabVIEW for Windows (National Instruments, 1993), which was the software platform used for ICS development, offers a graphical programming environment for constructing application software for the PC.

LabVIEW for Windows is a 32-bit integrated programming environment made for the 16-bit Microsoft 3.x operating system. This discrepancy caused incompatibility between LabVIEW programs and custom Windows DLLs that contained heavy computations inappropriate for programming in a graphical programming environment. To resolve this incompatibility, a third-party program Downshift (Viewpoint Software Solutions, 1993) was used as a bridge between the two environments. The future version of Microsoft Windows (Windows Chicago or Windows 95) is planned to be a full 32-bit operating system, which will make the use of the bridging program unnecessary.

Appendix B : THEORETICAL BASIS FOR MACHINE LEARNING TECHNIQUES

ADAPTIVE LOGIC NETWORKS (ALNs)

Two versions of Adaptive Logic Networks were evaluated and implemented in the design of FES-control system presented in this thesis:

1. ALN V.2 based on learning algorithms embodied in the program Atree 2.7[1], which produces adaptive logic networks, dealing with binary numbers. The learning algorithm is implemented in the form of computer program "WALKON" described in Appendix C, which is based on the Adaptive Boolean Logic Element (ABLE) patented by W.W. Armstrong[2].

2. ALN V.3 based on learning algorithm Atree 3.0[3], which produces adaptive logic networks, dealing with continuous quantities instead with binary numbers. It is implemented in the form of a dynamic-link library (DLL) of computer functions for Microsoft Windows and it is based on more general use of adaptive linear threshold elements, recently developed by W.W. Armstrong and M.Thomas.

ATREE version 2.7

The learning algorithm called Atree 2.7 is a computer program which implements the ABLE principles in approximating Boolean functions. Although it deals with binary numbers and uses Boolean operators to process binary information, the use of the program is not restricted to the binary information. Through the use of various quantization and encoding techniques for converting continuous quantities into binary numbers it is possible to process any information with this algorithm.

Adaptive Boolean Logic Element (ABLE)

The ABLE is defined as follows:

"A digital logic circuit, ..., of which an interconnected plurality together with their connections to a control unit comprise a learning machine which synthesizes a Boolean function of n variables as the result of a training procedure.

Each element may operate as a two-input, one-output combinational circuit of one of four logical types, where the type is determined by the current internal state of the element. All four types are such that a $(ZERO, ZERO)$ input pair gives rise to a $ZERO$ output, and a (ONE, ONE) input pair gives rise to a ONE output, while the particular operation realized is determined by two function value units which compute suitable outputs under the $(ONE, ZERO)$ and $(ZERO, ONE)$ input pairs to the element.

---

[1]ATREE 2.7 simulator, available from ftp.cs.ualberta.ca in file pub/atree/atre27.exe (binary mode), including C++ source code and extensive on-line help.

[2]W.W. Armstrong, Adaptive Boolean Logic Element, U.S. Patent 3934231, Feb. 28, 1974 (fillings in various countries), assigned to Dendronic Decisions Limited, 3624 - 108 Street, Edmonton, Alberta, T6J 1B4, Tel. (403) 4381103. N.B. Expired January 1993.

[3]W.W. Armstrong and M.M. Thomas, Dendronic Decisions Atree 3.0 beta release 1, ALN Theory, A practical Guide to Aproximating Relations

Three different species of element are described differing in their computation of heuristic responsibility. The 'global search' species is useful when the function to be synthesized is not constant; there is a convergence theorem related to this species. The 'latest error' species is also useful for applications where the function to be synthesized may be a constant; but this type does not obey a known convergence theorem. The 'hill climbing' species is useful principally to improve an existing approximate synthesis.

Means are provided for setting and reading out the functions realized by the elements in a tree-like network in order to facilitate storage and transmission of synthesized functions."

Due to the state of computer technology at the time of invention, the adaptive Boolean logic element was invented and patented as a learning machine whose adaptation process occurred in its hardware. This invention provided significant improvement to existing machine learning techniques.

Background of the invention of ABLE

Let us consider an idealized model for adaptive classification (Armstrong, 1974). We are given a training set of stimuli, which are represented as n -tuples of 0's and 1's, and a partition of this set into a 0-class and 1-class representing the desired classification on the training stimuli. We execute the following random search procedure to find a Boolean tree function which correctly classifies all members of the training set:

1. Let $L = 0$.

2. Consider the balanced binary tree, such as one illustrated in Figure B.1, having L layers of bifurcations ($2^L - 1$ non-leaf nodes, and 2^L leaf-nodes); to each leaf-node randomly assign one of the n Boolean variables or its complement (with multiple 'connections' allowed).

3. To each non-leaf node randomly assign one of the four non-constant increasing Boolean functions of two variables: *AND*, *OR*, *LEFT* or *RIGHT*, and interconnect them according to the tree.

4. Repeat step 3. a certain number of times, stopping any time the currently synthesized Boolean tree function of n variables is satisfactory, i.e. computes the correct class of any stimulus in the training set when the stimulus is presented at the n input variables.

5. If no satisfactory function has been obtained, go to step 2., choosing a new connection pattern to the variables and complements at the leaf-nodes, unless this has been tried a certain number of times without success.

6. If no satisfactory function has been obtained, increase L by 1 and go to step 2.

7. Stop. The current function of n variables correctly classifies all members of the training set.

If a satisfactory function has been found, it may be used to classify any input n-tuple, whether in the training set or not, by simply applying that n-tuple to the n input variables. Armstrong proved that, given the number n, a network of elements with appropriate connections to n logical signals and their complements could, in theory, be constructed in such a way that any of the 2^{2^n} Boolean functions of n variables would be attained for at least one state (Armstrong, 1976). The resulting machine would thus be a universal function-learner for n variables. Such a machine was described by Halpern in 1966 (Halpern, 1966). The problem with Halpern's machine grew with the number of variables n, so that for n in the hundreds or thousands, the practical realization of the machine was very difficult, due to the great number of possible states and therefore parts required

(at least 2^n). It was clear that for practical applications this idealized procedure must be replaced by a much faster heuristic procedure.

Fortunately, many useful functions of a large number n of variables are of very low sensitivity compared to most Boolean functions of n variables. The term sensitivity is used with regard to producing changes in the output of the learning machine when certain number of inputs is changed. A sensitivity is approached theoretically in Bochmann and Armstrong (1974) and will not be elaborated in more details here.

There were several designs for insensitive functions before the design of ABLE:

- Perceptron of F. Rosenblatt (Minsky and Papert, 1969),
- The networks of Artrons of R.J. Lee (Lee, 1967),
- The Slam networks of I. Aleksander (Aleksander, 1970), and
- The trainable digital apparatus of W. Armstrong (Armstrong, 1971).

The central part of Perceptron, where the learning occurs, is capable of realizing a certain class of, so-called, linearly separable classification problems. Its operation is based on a Perceptron convergence theorem which states that certain training algorithm will lead to a state in which the output signal is always the specified response to the input signals. A fundamental difficulty with the Perceptron system is that the class of linearly separable functions of n variables is extremely small so that the power of this system must be augmented by finding task-specific transformations of the data before they are applied to the function-learner. The advantage of the ABLE over Perceptron is that such external augmentation was not needed in systems based on ABLEs.

It is important to notice that at the time the ABLE was patented, the author stated that "the use of several layers of linear-threshold learning devices to attain a larger class of functions has not been very successful up to the present time since no satisfactory training algorithm was known for such networks". The ABLE represented a solution to this problem, though for a very special kind of linear threshold element. Atree version 3.0, which uses non-restricted linear threshold elements as an adaptive elements and which is described in the next section, represents more general solution to the problem.

The design of ABLE had advantages over designs listed above with regard to the capacity of learning and insensitive generalization. The restriction of the class of Boolean functions of two variables realizable by the elements of a binary tree network to four nonconstant increasing functions of two variables is crucial. These functions are:

$$AND(x,y) = xy$$
$$OR(x,y) = x + y$$
$$LEFT(x,y) = x$$
$$RIGHT(x,y) = y$$

All four functions have increasing nature which equates the output signal of an element in the network to the output of the network whenever that element is responsible, i.e. whenever changing its output would change the output of the network. A convergence theorem states that a synthesis of a specified function will be obtained by means of a certain algorithm for assigning these four functions to nodes of a binary tree provided:

- a synthesis of the function exists, and
- the components of the n-tuples are statistically independent under the distribution P.

This theorem is proved in the paper by Armstrong and Bochmann (1972), but the algorithm based on the theorem is not sufficiently powerful for practical applications, and it is therefore replaced by a statistical procedure based on the concept of "heuristic responsibility" of an element when a certain n-tuple is at the input of the network. The implementation of this concept significantly speeds up the training process and makes practical applications based on ABLE possible.

The ALN adaptation algorithm

The text in this section is mainly adapted from Armstrong and Gecsei (1979). To explain the ALN adaptation algorithm implemented in Atree version 2.7, the focus will be on the functions causing two-input elements (nodes) arranged in a binary tree to assume particular states in which they each realize some Boolean function of two variables. The leaves of the tree are connected to input variables either in a one-to-one fashion (the disjoint case) or with multiplicity of connections going to the same input variable and its inverse (the non-disjoint case). The functions which can be realized by any node of the binary tree are AND, OR $LEFT$ and $RIGHT$, i.e., $g(x,y) = xy$, $x+y$, x, y, respectively. Important to note is that the set of node functions proposed consists precisely of all non-constant increasing Boolean functions of two variables[4]. The tree composition of such node functions is also increasing and nonconstant. To prevent this being a real restriction, inversions are allowed in the connections of the leaves of the tree to the input variables[5]. Clearly, a sufficiently large nondisjoint tree can compute any Boolean function.

The main reason for using tree functions of the above restricted type is because they preferentially realize, for any fixed size and shape of tree, Boolean functions which have the property that an output value tends to remain unchanged when small perturbations are applied to the input variables. This property of tree functions has been referred to as *insensitivity* (Bochmann and Armstrong, 1974) and it provides good extrapolation or generalization on the test data set. It is particularly pronounced in *Binary Tree Functions* (BTF), and within that class more so when the node functions are restricted to be AND, OR, $LEFT$ and $RIGHT$. This holds for both disjoint and non-disjoint types of the tree.

Theoretically (Armstrong W.W. and Godbout G.; 1974, Bochmann and Armstrong, 1974), even a randomly chosen binary tree function compatible with the training data is a good basis for extrapolation. The adaptation algorithms can be regarded as making the random choice in an accelerated manner.

The most difficult problem in designing en effective adaptation procedure is to determine which nodes to change during adaptation. This is the basic "credit-assignment problem" referred to by Minsky in (Minsky, 1961). This solution involves recursive computation of "heuristic responsibility" signals which inform the two descendants of a node whether or not they are candidates for changing state at a given adaptation step.

Definition of binary tree functions

Consider a binary tree composed of the set K of *nodes*. Each node k has attached to it one of the following:

---

[4]If a zero at an input is changed to a one, an increasing function will never change its value from one to zero.

[5]There is no need to have inversions between elements of the tree, since by simply passing from AND to OR or *vice versa*, one can move an inversion from the output of an element to its two inputs and iterate until the only inversions left are at the input variables.

- two *successor* nodes k_l and k_r, or
- one successor k_l or k_r, and one connection to one of the input variables $x_1, x_2, \ldots, x_m$, or
- two connections to the input variables.

Any connection to an input variable may contain an inverter. Figure B.2 illustrates an example of such a tree.

Node k_1 will be called *descendent* of k_2 if $k_2 = k_1$ or k_1 is a successor of a descendent of k_2. Each node is a two-input, one-output combinational circuit receiving Boolean input values from its successors or from the input variables or their complements. Let $G = \{AND, OR, LEFT, RIGHT\}$ be the four non-constant increasing Boolean functions of two variables. A mapping $g: K \rightarrow G$ will be called an *assignment*. The function assigned by g to node k will be denoted by g_k. Figure B.2 and Table B.1 illustrate these concepts.

Table B.1. Definition of four node functions.

f_{kl}	f_{kr}	$g_k \rightarrow$	*AND*	*LEFT*	*RIGHT*	*OR*
0	0		0	0	0	0
0	1		0	0	1	1
1	0		0	1	0	1
1	1		1	1	1	1

For any fixed assignment g, the value at the output of the root node is $f(x_1, x_2, \ldots, x_m)$; g is said to *realize* the function f.

A Boolean function $h(x_1, x_2, \ldots, x_m)$ is *realizable* by a given tree including a given connection pattern to the variables, if and only if there is an assignment such that $f \equiv h$. Such an h will be called a *Binary Tree Function* (BTF) for that tree.

Decomposition of disjoint binary tree function

As pointed out previously, any Boolean function is a BTF for some tree. If the connections are one-to-one and contain no inversion, we shall speak of a *Disjoint Binary Tree Function* (DBTF). That is, each variable of DBTF is connected to exactly one leaf node of the tree, as for example in Figure B.3. Without the loss of generality, we may assume that the tree used is *balanced*. There are n levels, with level n containing only the root node, and level 1 containing 2^{n-1} nodes. The tree has 2^n inputs connected one-to-one to 2^n variables.

Due to their fixed and known underlying structure, every DBTF of 2^n variables has a simple disjoint decomposition of the form

$$f_k = \begin{cases} g_k(f_{kl}, f_{kr}), & \text{for } k \text{ in levels } 2, 3, \ldots, n \\ g_k(x_{kl}, x_{kr}), & \text{for } k \text{ in level } 1 \end{cases}$$

where $x_{kl}(x_{kr})$ is the variable connected to the left (right) input of a leaf node.

Take as an example the tree in Figure B.3. The DBTF realized by this tree has the form

$$f = f_7(x_1, x_2, \ldots, x_8) = g_7\left(g_5\left(g_1(x_1, x_2), g_2(x_3, x_4)\right), g_6\left(g_3(x_5, x_6), g_4(x_7, x_8)\right)\right). \quad \text{(B.1)}$$

Thus, in order to determine if a given function $h(X)$ is a DBTF for the tree of Figure B.3, one has to establish whether a decomposition of h in the form (B.1) exists. If it does, then $g_k\ (k \in K)$ specifies the assignment realizing h; otherwise h is not a DBTF for this tree. It has been shown (Bochmann and Armstrong, 1974) that such an assignment is unique if and only if h depends on all variables in X.

To be able to understand better the adaptation algorithm, a decomposition method introduced by Curtis (1962) is described briefly in the following section. The starting point is the tabular representation of the function $h(X)$ to be decomposed; the end product is $2^n - 1$ *decomposition charts* (DC) (one DC for each node function g_k). Decomposition charts are the principal means for the explanation of adaptation algorithms later in the paper.

Let X be the set of variables $x_1, x_2, \ldots, x_m$ and $X_k \subseteq X$ the set of variables connected to the descendants of node k. Let $X_k' = X - X_k$, i.e., X_k' represents the set of variables not connected to the descendants of the node k.

A DC is a table of four rows corresponding to combinations of input values f_{kl}, f_{kr} of node k (x_{kl}, x_{kr} if k is a leaf node) and $2^{X_k'}$ columns, one for each possible value of X_k'. The entries in the table are the prescribed values of h.

Take as an example the function $h(X) = x_1x_2 + x_3 + x_4$, which is a DBTF. Its tabular representation is shown in Table B.2 and the corresponding two-level tree in Figure B.4. The DC for leaf nodes is just a simple rearrangement of Table B.2 as seen in Table B.3.

Table B.2. Truth table for $h = x_1x_2 + x_3 + x_4$.

x_1	x_2	x_3	x_4	h	f_1	f_2
0	0	0	0	0	0	0
0	0	0	1	1	0	1
0	0	1	0	1	0	1
0	0	1	1	1	0	1
0	1	0	0	0	0	0
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	1	0	1
1	0	1	1	1	0	1
1	1	0	0	1	1	0
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

Table B.3. Decomposition charts for nodes 1 (A), 2 (B), and 3 (C).

(A) $\{x_3 x_4\} = X_1'$

x_1	x_2	00	01	10	11
0	0	0	1	1	1
0	1	0	1	1	1
1	0	0	1	1	1
1	1	1	1	1	1

(B) $\{x_1 x_2\} = X_2'$

x_3	x_4	00	01	10	11
0	0	0	0	0	1
0	1	1	1	1	1
1	0	1	1	1	1
1	1	1	1	1	1

(C) $X_3' = 0$

f_1	f_2	
0	0	0
0	1	1
1	0	1
1	1	1

In order to construct the DC for g_3, the assignments to g_1 and g_2 should be determined. This can be done by analyzing the columns appearing in Table B.3.(A) and (B). In each DC there are *constant* and *non-constant* columns (CC and NC). Constant columns (e.g., column 11) contain only one value (0 or 1) for all combinations of signals belonging to X_k. The meaning of CC is that at the given value of X_k' (shown above the CC), node k has no influence on (responsibility for) the output. This is so since all four possible functions $g_k \in G$ are non-constant. Thus, CC in a DC give no indication for the choice of an assignment to g_k. The non-constant columns, on the other hand, indicate that (at those values of X_k') node k does influence (is responsible for) the value of h. In effect, $f_k = h$ in all positions of an NC.

Therefore, for any DBTF

a) all NC in a DC must be identical among themselves, and

b) all NC in a DC must be identical to one of the four columns in Table B.1 (i.e., correspond to a $g_k \in G$).

This immediately yields in the example presented in Figure B.4 $g_1 = AND, g_2 = OR$. If there is no NC in a DC, any assignment to g_k can be made.

The next step in the evaluation of the example is to construct the DC for g_3, shown in Table B.3.(C). The entries in this DC are obtained using columns f_1 and f_2 in Table B.2. Since there are generally more lines than one with the same values of f_1, f_2 (unlike the construction of DC for leaf nodes), it is conceivable that both $h = 0$ and $h = 1$ are found as a particular entry in the

DC. Such would be the situation if $h(X)$ were not a DBTF on the given tree, e.g., $x_1x_2x_3 + x_4$. However, it can be easily seen that this situation is impossible when all DC's on the lower level of the tree satisfy conditions a) and b). Thus, $g_3 = OR$, which concludes the example of determining g_k by the application of decomposition charts.

To recapitulate, this decomposition method is an inherently sequential procedure where the assignments obtained for a lower level are used to determine those for the next higher level. The existence of all $g_k \in G$, $k \in K$ (conditions a) and b)holding for all DC's) is the necessary and sufficient condition for a function to be a DBTF on the given tree.

Definition of true responsibility

Let K be a binary tree, g an assignment realizing $f(X)$, and ξ a particular input vector (value of X). The notion of responsibility of a node k, as a situation where $f_k = f$ in all position of a NC, was introduced in the preceding section. An equivalent description of this would be to state that k is responsible if and only if changing the value of f_k would change the value of f. (One can imagine an "experimental" determination of responsibility by cutting the line f_k, injecting signal values 0 and 1, and seeing if the tree output f follows the injected values.) For fixed g, the responsibility depends, of course, on the actual input vector ξ. The root node is evidently always responsible.

This concept is formalized as follows:

1. $\rho_t(\xi) = 1$, for all ξ

2. $\rho_{kl}(\xi) = 1$, iff $\rho_k(\xi) = 1$ and one of the following three conditions holds:

   a) $g_k = LEFT$ or

   b) $g_k = AND$ and $f_{kr}(\xi) = 1$ or

   c) $g_k = OR$ and $f_{kr}(\xi) = 0$.

3. $\rho_{kr}(\xi) = 1$, iff $\rho_k(\xi) = 1$ and one of the following three conditions holds:

   a) $g_k = RIGHT$ or

   b) $g_k = AND$ and $f_{kl}(\xi) = 1$ or

   c) $g_k = OR$ and $f_{kl}(\xi) = 0$.

Take as an example the function $f(X) = x_1x_2 + x_3 + x_4$ in Table B.2. We recall that $g_1 = AND$, $g_2 = OR$ and $g_3 = OR$. Setting $\xi = 1100$ we find $\rho_1 = 1, \rho_2 = 0, \rho_3 = 1$; setting $\xi = 0110$ yields $\rho_1 = 0, \rho_2 = 1, \rho_3 = 1$.

It can be seen that $\rho_k(\xi)$ actually depends only on those g_i such that i is *not* a descendent of k and on those components of ξ which are in $X_k' = X - X_k$. This is equivalent to what was said previously about responsibilities: node k is truly responsible precisely for the non-constant columns in its decomposition chart. A generalized notion of responsibility is essential to the operation of some adaptation algorithms presented in the following sections.

Adaptation algorithms

With the final goal of implementing these algorithms in fast hardware, two approaches are possible for adaptation of the binary trees: first, the node assignments can be determined in the external computer and downloaded into the tree of fixed connections, and the second, the determination of assignments can be done within the node elements themselves, resulting in distributed-intelligence system. The tree nodes would be capable of gathering information directly from their inputs and adjusting their behavior (assignment) according to the desired values of $h(X)$. The nodes then become *adaptive logic elements* whose processing power and interconnections with other elements are relatively limited. Adaptation algorithms are considered in the sense of actions executed by such adaptive elements. In studying some of the algorithms the results and notions (e.g., decomposition chart, responsibility) seen in the preceding sections will be used. The advantage of the second approach is much greater speed of learning than is possible with a general-purpose computer. The main disadvantage is the cost of special hardware.

Figure B.5 shows the system configurations envisaged by the above two approaches. To understand their operation, it will be useful to distinguish between the training set and the test set of input vectors. The former is used to obtain the node functions; the latter represents subsequent classification tasks on input vectors. Actual adaptation occurs only while elements of the training set are presented (with the desired value of h); the node functions are fixed during the classification (testing) period. The size of the training set depends on the application.

In the adaptive system in Figure B.5(B), each node k receives from its predecessor a signal s_k, called *heuristic responsibility*, indicating whether k is considered to be responsible for the actual output f and "enabled" for adaptation during the presentation of the current input vector. Heuristic responsibility is usually not equal to true responsibility defined in previous section.

Adaptation algorithms for DBTF

The method of decomposition charts is not directly applicable to a system with adaptive nodes because it would be unrealistic to assume that a node has enough memory to hold the (possibly very large) decomposition chart.

The method is, thus, modified so that while the input vectors are presented, the internal logic of a node k gathers information relevant to its assignment, via the sequence of triplets $\left[f_{kl}, f_{kr}, h(\xi)\right]$ presented at k's input at every new input vector ξ. This accumulation of information within k can be done essentially with the aid of a few counters and associated logic or arithmetic circuits.

Adaptation algorithms can be viewed as an attempt to determine the nature of non-constant columns in a DC. The main problem is that unless the sequencing of vectors is controlled by the algorithm, it is difficult for a node to determine which column has occurred. On the other hand, the lines of the DC can be readily distinguished by the values of (f_{kl}, f_{kr}). Thus, the processing done by the nodes will have to include some form of statistical analysis of lines (e.g., by counting the occurrences of $h(\xi) = 0$ and $h(\xi) = 1$ in each line).

There are two types of algorithms:

a) All columns (NC and CC) are used in the search for the correct assignment. Such algorithms will be called *homogeneous* since all nodes of a tree level adapt independently from all others except their descendants. The remainder of this section contains a typical example. The algorithm is demonstrated to work for any DBTF on the given tree.

b) An attempt is made to recognize the NC's of a node using the states of nodes which are not descendants, and these "tentative" NC's are used to find the node assignment. The convergence of these *responsibility-based* algorithms has not been proved, but they have been demonstrated to work much better than homogeneous algorithms in approximating non-realizable functions in the non-disjoint case. Two such algorithms are presented in the next section.

The following observation are made:

Observation 1: Since in a DC of a DBTF the NC's are all the same, there can be maximally two different kinds of lines (e.g., the DC in Table B.3(A) has lines 0111 and 1111). Designate these two types of lines in a DC by Q and R, defining the line corresponding to the input pair (00) to be of the type Q. When all lines are of the type Q, there is no NC. Otherwise, the line corresponding to the input pair (11) is of type R.

Observation 2: The ones in lines Q are a subset of the ones in lines R. Therefore line types can be obtained from the numbers of ones and zeros in the four lines of the DC.

Observation 3: Lines of types Q and R can be distributed in a DC in one of five possible ways, as shown in Table B.4. The resulting node functions are in the bottom line.

Table B.4 Line types in DC and the determination of node functions

	column →		1	2	3	4	5
line ↓ index	f_u	f_v	$K_1>K$ $K_2>K$	$K_1>K$ $K_2<K$	$K_1<K$ $K_2>K$	$K_1<K$ $K_2<K$	$K_1=K$ $K_2=K$
0	0	0	Q	Q	Q	Q	Q
1	0	1	R	R	Q	Q	Q
2	1	0	R	Q	R	Q	Q
3	1	1	R	R	R	R	Q
	g_t →		OR	RIGHT	LEFT	AND	any

Observation 4: Direct counting of ones (i.e., occurrences of $h(\xi)=1$) by nodes would give the actual number of ones in a line only in leaf nodes. In non-leaf nodes a DC entry generally corresponds to several vectors ξ (as in the example of Table B.2); hence different number of ones may be obtained even in two lines of the same type. However, it is easy to see that if all vectors ξ are presented exactly once, the *fraction* of observed ones and zeros is the same for all occurrences of a given line type.

Consider now only the cases where both line types exist. Let u_i and $z_i, i \in \{0,1,2,3\}$, be the numbers of ones and zeros observed in line i and $K_i = u_i / z_i$. We observe that line zero being always Q and line 3 always R,

$K_0 < K_3$ (K_3 may be infinite).

Further, K_1 and K_2 must be equal to either K_0 or K_3. Therefore, for some $\alpha > 0, \beta > 0$ we can write

$$K = \frac{\sum_{i=0}^{3} u_i}{\sum_{i=0}^{3} z_i} = \frac{\alpha u_0 + \beta u_3}{\alpha z_0 + \beta z_3}.$$

Lemma: Let $K_0 = u_0 / z_0 < K_3 = u_3 / z_3$, with $z_0, u_3 > 0$, and $u_0, z_3 \geq 0$. Then

$$K_0 < K < K_3. \tag{B.2}$$

Proof: We have

$$u_0 z_3 < u_3 z_0$$

hence

$$\beta u_0 z_3 + \alpha u_0 z_0 < \beta u_3 z_0 + \alpha u_0 z_0$$

which leads to

$$\frac{u_0}{z_0} < \frac{\alpha u_0 + \beta u_3}{\alpha z_0 + \beta z_3}.$$

We prove similarly the second part of the inequality.

Inequality (B.2) can be used by an adaptive algorithm to determine the line types and thus the node functions, since for $i = 1, 2$

$K_i < K$ iff line i is of type Q

$K_i > K$ iff line i is of type R.

One algorithm employs six counters C_{01}^0, C_{01}^1, C_{10}^0, C_{10}^1, C^0, C^1 per node, used to determine $z_1, u_1, z_2, u_2, \sum_{i=0}^{3} z_i$ and $\sum_{i=0}^{3} u_i$ respectively. These values will be accumulated during training of a particular layer, consisting of one presentation of every vector ξ together with the desired value of $h(\xi)$. Comparing K_1 and K_2 to K gives the required node functions, as shown in Table B.4.

Correct adaptation depends, of course on correct subtree values (f_{kl}, f_{kr}); therefore, in an n-level tree, n training periods are necessary, first training the leaf nodes, and then those in the next layer, etc. This was summarized as algorithm A1.

A1 (Homogeneous adaptation)

1) Set all counters in all nodes to zero

2) DO for levels $j = 1, 2, \ldots, n$

3) DO (possibly in parallel) for all nodes k in level j (for a sequence of all vectors)

DO for every input vector :

calculate the outputs of all descendants of k starting at the leaves of the tree

change the state of node k as follows:

f_{kl}	f_{kr}	$h(\xi)$	Action
0	0	0	$C^0 \leftarrow C^0 + 1$
0	0	1	$C^1 \leftarrow C^1 + 1$
0	1	0	$C^0 \leftarrow C^0 + 1, C^0_{01} \leftarrow C^0_{01} + 1$
0	1	1	$C^1 \leftarrow C1 + 1, C^1_{01} \leftarrow C^1_{01} + 1$
1	0	0	$C^0 \leftarrow C^0 + 1, C^0_{10} \leftarrow C^0_{10} + 1$
1	0	1	$C^1 \leftarrow C^1 + 1, C^1_{10} \leftarrow C^1_{10} + 1$
1	1	0	$C^0 \leftarrow C^0 + 1$
1	1	1	$C^1 \leftarrow C^1 + 1$

END;

Determine g_k as in Table B.4.

END;

END;

A more general treatment of this algorithm is given in Armstrong and Bochmann (1972).

Adaptation for non-disjoint trees and non-realizable functions

The majority of practical pattern classification problems lead to Boolean functions of a great number of variables. Generally, such functions can not be realized by any disjoint tree, nor by any given non-disjoint tree. While it is clear that such non-realizable functions can not be synthesized exactly, it is important to see if *approximate* synthesis is possible.

In this section two algorithms are demonstrated which successfully approximate on fixed non-disjoint trees.

Disadvantage of Homogeneous Adaptation algorithm in non-disjoint cases is its homogeneity: all nodes in the tree (or in a level) adapt in ignorance of what is happening in other nodes except descendants and regardless of whether they are responsible in a given situation or not. It was concluded that similarly to other complex learning algorithms, it is essential not to vary too many parameters at a time; chances of successful adaptation will increase if a subset of nodes can be found whose function is "important" for the synthesis of the desired output for a given vector, adaptation being done only on this subset while the state of other nodes is frozen.

A natural indicator of importance of a node in the above sense appears to be its responsibility $\rho_k(\xi)$ introduced earlier. As it was defined, the responsibility of the left (right) successor of node k depends on the opposite right (left) input of k, on g_k and on whether k is responsible. Thus the responsibility of any node can be determined by its predecessor, which is expressed by the descending signal lines s_k in Figure B.5(B).The information on s_k is designated heuristic responsibility (which may be different from the true responsibility ρ_k as in A3). Heuristic responsibility can be viewed in two ways: first as an attempt to approximate ρ_k in a correctly trained tree; second, it is a signal that enables adaptation in node k.

Responsibility-based algorithms are those which allow only responsible nodes to adapt at any time during the training. It's been shown in previous sections that $\rho_k = 1$ in an indication of node k being in a NC. It can be shown that the intersection of any line with all NC in the decomposition chart of a DBTF can contain only zeros or only ones. Therefore, *if* there was a way of knowing the

correct value of ρ_k even in an untrained tree, and *if* the functions were limited only to DBTF, two flip-flops (one for each line 01, 10), set by the value of $h(\xi)$ in the first NC encountered, would suffice for the determination of g_k. Since none of this conditions is generally valid, a reasonable choice seems to be to use one up/down counter in each line (01 and 10), indicating in its sign bit whether ones or zeros are in majority. Up (down) counting occurs only if the responsibility signal indicates that the column is assumed to be a NC.

Defining $s_k = \rho_k$ leads to adaptation algorithm A2, where the nodes contain sufficient logic to generate ρ_k and two counters C_{01} and C_{10}. The instantaneous state of these counters determines at all times g_k.

A2 (True responsibility Algorithm)

1) Set all counters to zero

2) DO for a "long" random sequence of input vectors ξ:

a) calculate the outputs of all nodes starting at the leaves of the tree

b) determine $s_k(\xi) = \rho_k(\xi)$ for every node in the tree as in Section, starting at the root

c) change the states of the nodes as follows

$s_k(=\rho_k)$	f_{kl}	f_{kr}	$h(\xi)$	Action
1	0	1	0	$C_{01} \leftarrow C_{01} - 1$
1	0	1	1	$C_{01} \leftarrow C_{01} + 1$
1	1	0	0	$C_{10} \leftarrow C_{10} - 1$
1	1	0	1	$C_{10} \leftarrow C_{10} + 1$

d) set g_k to

$$\begin{aligned} AND \quad & if\ C_{01} < 0, C_{10} < 0 \\ LEFT \quad & if\ C_{01} < 0, C_{10} > 0 \\ RIGHT \quad & if\ C_{01} > 0, C_{10} < 0 \\ OR \quad & if\ C_{01} > 0, C_{10} > 0 \end{aligned}$$

END;

This algorithm (implemented with limits on C_{01} and C_{10}) is quite unsuccessful; it has the drawback that generating ρ_k in the indicated way does not yield in an untrained tree its "correct" values (i.e., ρ_k as they would appear in an already correctly trained tree). Having these correct true responsibilities would be, of course, the ideal guide for efficient training. A2 tries to use the incorrect responsibilities and hopes that after some amount of training they will bring about some improvement in node assignments which can, in turn, cause more correct (statistically) responsibilities, etc. It can be shown that any single wrong node assignments in an otherwise well-trained tree will be readily corrected by A2. On the other hand, A2 has the tendency to "lock in" local optima even if better solutions are attainable through simultaneous changes in several node assignments at once.

A significant improvement over A2 can be achieved by excluding from the computation of s_{kl}, s_{kr} the influence of the (possibly wrongly assigned) g_k and replacing it by an assumption based on the (correct) value of desired output $h(\xi)$. Define the situations $(f_{kr}=0, h(\xi)=1, s_k=1)$ and $(f_{kr}=1, h(\xi)=0, s_k=1)$ as errors (made by k_r for the input ξ) of types 0 and 1, respectively. Take the point of view that in these situations the left successor k_l must be heuristically responsible $(s_k=1)$ and should adapt accordingly to eliminate the error. Further, node k_l will also be held responsible whenever the value of f_{kr} (0 or 1) occurs that caused the *latest error* (i.e., until the occurrence of an error of the opposite type). By similar reasoning we determine independently s_{kr} based on the latest error in f_{kl}.

A little reflection reveals that this way of determining heuristic responsibility is equivalent to *assuming* that on an error of type 1, g_k becomes AND and stays that way until an opposite error is encountered which changes the assumed g_k to OR. These assumptions are made simultaneously by the logic for generating s_{kl} and s_{kr} and thus may be contradicting each other at any time.

In a practical implementation each node would contain two flip-flops E_L and E_R to remember the type of latest error observed in f_{kl} and f_{kr}. Thus $E_L \leftarrow 1$ when $f_{kl}=1, h(\xi)=0$,and $s_k=1$; $E_L \leftarrow 0$ when $f_{kl}=0, h(\xi)=1$, and $s_k=1$. E_R is controlled similarly by f_{kr}. The values of E_L and E_R can be initialized arbitrarily. Now we can formulate the "latest error" algorithm:

*A3 (*Latest Error Algorithm*): As A2 except for step 2b):*

...

2b) Set $s_k=1$ for k the root node. For every node, starting at the root update E_L and E_R if $s_k=1$; and

determine s_{kl} for the left successor of node *k* as $s_{kl}=s_k \wedge (f_{kr}=E_R)$

and for its right successor as $s_{kr}=s_k \wedge (f_{kl}=E_L)$

...

While it is difficult to give precise arguments or proofs justifying A3, a simple example will illustrate its advantage over A2.

Lets consider a sequence of nodes arranged as in Figure B.6. Considering $x_3x_2x_1x_0$ as a four-bit binary number x, the task for the tree is to learn the function f where $f(x)=1$ if and only if $x \geq y$ for some fixed threshold y. Lets choose $y=1010$ (ten in binary) for concreteness. The input x_3 is an "error" in the sense of A3 only for the inputs $x=1000$ and $x=1001$, when the desired network output is 0. Hence as soon as one of these vectors is presented during adaptation, the latest error E_L for x_3 is fixed at the value 1. At this adaptation step, and those following, the right descendent, node 2 is heuristically responsible if and only if $x_3=1$. This in effect, imposes the task of learning the three-bit threshold $x_2x_1x_0 \geq 010$ on the right subtree of node 1, since for inputs with $x_3=0$, the right subtree does not adapt. By repeating the argument for node 2, it can be seen that the right subtree of node 2 is responsible if and only if node 2 is heuristically

responsible $(x_3 = 1)$ and x_2 equals the latest error on x_2. Errors for x_2 occur for $x_3x_2x_1x_0 = 1010$ and 1011 when the desired network output is 1: so node 3 is heuristically responsible if and only if $x_3 = 1$ and $x_2 = 0$. Thus the task this imposes on node 3 is to learn $x_1x_0 \geq 10$, which leads to node 3 becoming a $LEFT$ function, independently of what function is realized by nodes 1 and 2. Next, whenever node 2 is heuristically responsible $(x_3 = 1)$ and when its input pair is 01 (cases x=1100 and 1101), ,or when its input pair is 01 (cases x=1010 and 1011), a 1 output is desired for the network. This leads to node becoming an OR function independently of what is in node 1. Finally, whenever node 1 has inputs 10 or 01 the desired network output is 0, and node 1 adapts to become an AND function.

Practical applications of early Adaptive Logic Networks

The new approach to high-speed pattern classification was described by Armstrong and Godbout (1974). Functions were synthesized in form of Boolean trees formed of two-input, one-output elements capable of realizing AND, OR, $LEFT$ and $RIGHT$ operators. A theory is developed and computer simulation experiments are described involving applications ranging from pattern recognition to performing key-to-address transformations for information retrieval. The former type of application is one which critically depends on so-called "smooth extrapolation" or generalization on the data not used during the training, while the latter type does not necessarily require it. The practical applications were recognition of handwritten numerals, numerical taxonomy and medical diagnosis.

A typical result from this study was 91.7% correct classifications of the test set in the well-known problem of statistical classification of iris plants. The problem is defined as follows:

- the data describe 150 iris plants, 50 in each of three species,
- each plant has four measurements (petal length and width and sepal length and width),
- the goal is to train the network with a part of the data set (training set) and to achieve the best classification on the rest of the data (test set).

Since the measurements of the flower's dimensions are not usually expressed in binary numbers, the first step in applying tree networks to this problem was to find a way of representing the numerical measurements as bit vectors at the inputs to the tree. Although any binary coding would provide the required information, a better scheme using knowledge that these bits represent a numerical quantity was applied. First the range of each measurement was divided into 2^p subintervals such that the frequencies of occurrence of measurements in all of these were (roughly) equal. The choice of p is a question of judgment, but in any case when the ordered sequence of subintervals is represented by the ordered set of binary integers from 0 to $2^p - 1$ each bit of that representation carries one bit of information in the Shannon sense.

Another application of adaptive tree networks was in statistically more difficult task. The problem is defined as follows:

- the data describe 299 patients, having one of six diseases,
- each patient is presented with 11-dimensional Boolean vector corresponding to presence or absence of eleven symptoms,
- the task is to classify the subjects according to which one of six diseases they have.

In this example the adaptive tree networks in their original form could not match the best results obtained using classical statistics which were 52% and 46 % correct classifications for training on the whole data set or just on a half of it respectively. However, using three majority votes over

several tree networks to compute each bit resulted in 45.2% and 52% correct classifications after training on halves of the data set and test on the another half.

A majority votes proved to be valuable technique for improving accuracy of ALNs. Instead of having only one tree for every bit of the output, a number of trees can be trained for that task. These trees then vote (majority wins) on what the output bit should be. This helps to eliminate errors caused by peculiarities of individual trees.

Latest error adaptation algorithm (A3) was applied (Armstrong and Gecsei, 1979) to two generally nonrealizable functions, which are approximated by learned BTF. The first problem is defined as follows:

- the data consisted of 400 Boolean variables representing a half-tone image on a 20 x 20 grid
- the images were obtained by taking a vertical dividing line and generating black picture points to the left of this line with probability b_l and to the right of this line with probability b_r.
- the vertical dividing line can be in any of the 20 positions except numbers 5 and 15 from the left
- the task is to determine whether the dividing line is between positions 5 and 15 or not.

The best results were obtained when majority votes over several 1023 element tree functions were used. In the case b_l=0.35, b_r=0.65, the human eye has difficulties to detect the dividing line. 15 of 60 trained trees with the best results were selected and used in a majority vote system. They were able to correctly classify 85.7% images.

In the second example, character images were created by applying distortions on a set of synthesized 16 x 16 characters obtained from a text editing system. The distortions involved translation, rotation by an angle between and degrees, and inversion of individual black or white dots with probability (binary symmetric channel or BSC noise). There were several tasks ranging from recognition of single character to the recognition of several characters. Overall results showed that majority votes improved performance of the recognition system in all cases. Up to 15 of 60 trained trees were used in majority votes in the most difficult task resulting in 96.6% correct recognitions. Results also confirmed previously stated theoretical claims about insensitivity of the BTF to noise.

Majority votes

According to Armstrong and Godbout (1974), it is advantageous to compute the majority value over several BTF in order to obtain good generalization from training to test sets. In principle, larger tree, a larger training set, and a longer training period, would be capable of producing the same effect. Taking majority votes is an easy way to improve extrapolation based on sparsely distributed training data. It saves creating larger training set and adapting for longer time to obtain the same quality of performance by a single tree.

ATREE version 3.0

Atree 3.0 is based on more recent developments by W.W. Armstrong and M. Thomas (1994) and deals with continuous quantities instead binary numbers. In Atree 3.0, the logic trees containing AND and OR operators have been furnished with input operators in the form of Linear Threshold Elements (LTE). The direct consequence is that with the new ALN design it is possible to apply ALNs directly to real values and thus define *piecewise linear approximations* of functions. This approach is not restricted to approximating functions, but can approximate relationships of more

general types represented as sets of data points or samples, which is important for pattern recognition applications.

The fundamental difference between Atree 2.7 and 3.0 versions is the way ALNs are used to solve problems (Armstrong and Thomas, 1994). Instead of computing an output as a *function* of some inputs, ALNs in Atree 3.0 are used to represent *relations* among inputs. The Atree 2.7 "output", in the functional sense of the word, now becomes just another input to the ALN, which computes whether or not all of its input quantities are related in a certain way. This subtle yet fundamental change in paradigm allows much greater flexibility than traditional neural networks architectures:

- The inverse of the function can be derived from the learned relationship without retraining.
- Constraints on the relations are easy to enforce, which allows *a priori* knowledge to be added into the system before learning begins. This is vital for safety critical applications, where *design* of safe systems is easier than *testing* to see if a system is safe.

An ALN is used to approximate a relationship among n inputs/outputs in an n-dimensional space. Such a space is called a *region*, and the ALN is called *approximant*. Each dimension in the space is called a *variable* of the region. Variables and approximants in a region may have subregions (subsets of the space) which specify different constraints on the variables and approximant (see Figure B.7).

Approximants

There are two types of elements used to form relation approximants in a region:

- logic gates (AND and OR), and
- linear threshold elements.

The logic gates may have an arbitrary number of logical inputs (fanin), and produce a logical output. The LTE have real valued inputs (represented by IEEE 64-bit double precision floating-point numbers), and also produce logical output.

Logic Gates

Logic gates can be AND or OR and they can have an arbitrary number of logic inputs. An OR gate has logical output value of zero if and only if all of its inputs are logical zero. The equivalent linear expression for an OR gate with n Boolean inputs is $x_1 + x_2 + ... + x_n \geq 1$. An AND gate has a logical output value of one if and only if all of its inputs are logical ones. The equivalent linear expression for an AND gate of n Boolean inputs is $x_1 + x_2 + ... + x_n \geq n$

Linear Threshold Elements

The basic element of approximation is the linear threshold element, which is very similar to the Perceptron developed in the 1950's. The LTE performs additions, multiplications and a comparison to zero of the form $w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n \geq 0$. It has a logical output value of one if this inequality is satisfied and a logical value of zero otherwise. All input points $(x_1, x_2, ..., x_n)$ that satisfy the inequality (thus producing output equal to one) are on one side of the *hyperplane* $w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n = 0$, forming a half-space (see Figure B.8).

The LTE fits the data presented to it during training by performing a least-squared error fit, similar to linear regression. Thus, the most elementary ALN which has only one LTE is capable of doing linear regression, but with possibly less accuracy than standard statistical package.

General ALN structure

General structure of the ALN usually includes a tree of several layers of logic AND and OR gates that have linear threshold elements as inputs (see Figure B.9). The inputs to the linear threshold elements are the inputs and outputs of the problem. The half-spaces formed by the LTEs are joined by the logic gates in *intersection* (AND) and *union* (OR) operations. This allows a piecewise representation of data sets or functions.

The goal of training an ALN is to recognize that certain input/output combinations are in a relationship to each other, while some other are not. The way an ALN represents a function as a relation is as follows:

"All points that lie on and under the graph of the function $(x_1, x_2, ..., x_n)$ will be classified as logical one, while all other points are to be classified logical zero."

Implementation of the LTE on simple classification problems

An LTE can be used as a linear classifier such as one presented in Figure B.10. In this example an LTE sets the simple hyperplane which separates two sets of data points in multidimensional input/output space.

Implementation of the LTE on complex classification problems

In more complex classification problems, such as one presented in Figure B.11, the resulting ALN is an AND combination of two hyperplanes, so that $Y = Y_1 \bigcap Y_2$.

ALN approximation of non-linear functional relationships

In the case of problems involving non-linear functions, the *piecewise linear approximation* is introduced. The resulting ALN tree may have only one logic gate to process outputs from linear threshold elements. For convex-type functions an AND logic gate outputs the intersection or minimum of all linear threshold elements in a given region (see example in Figure B.12).

For concave-type functions an OR logic gate outputs the union or maximum of all linear threshold elements in a given region. For functions having both convex and concave parts, the resulting ALN tree must have at least two layers, one of each AND and OR logic gates (Figure B.13).

Controlling the fitting accuracy of the linear threshold elements

Using a value called learning rate, a linear threshold element will adjust its weights to correct some of the error in its fit of a data point during training. The higher the learning rate, the more it will correct itself to fit the point, but correcting too much may change the fit of the linear element to other data points it has previously been responsible for. It may lose its responsibility for some points (responsibility which will be assumed by other linear elements), and it may gain responsibility for fitting new points that it hasn't seen before.

Unlike a standard linear regression that fits a fixed set of data points, a linear threshold element has a constantly changing set of points that it is trying to fit. Lowering the learning rate slows down adaptation, but makes the fit more accurate. The closer to zero the learning rate is, the more accurate the least-squares fit.

Constraints and ALNs

Having elements with adaptive weights only at the bottom level of an ALN, and knowing that there is only one linear threshold element responsible for an input point at any time, it is easy to place

constraints on the ALN relation approximant that will force it to learn only relations of a certain type and with a certain accuracy. This is equivalent to including an expert knowledge in the training process which can drastically speed up learning of relations, as well as provide an element of safety in design. Constraints on the ALN can be of the following types:

Range constraint which restricts possible values that an input of an ALN can take. This constraint is used to help decide responsibility when the input space is partitioned into several smaller regions, each with its own set of constraints on variables and approximants.

Epsilon constraint which helps to define the maximum desired accuracy of an approximant in a part of the input space. The learning algorithms will not waste time trying to adapt weights on variables that are already being fitted to a precision within the epsilon tolerance.

Monotonicity constraints are used to restrict the possible values that a weight on a variable can take in a linear threshold element. Monotonicity can be:

- free - in which a weight is allowed to vary freely;
- decreasing - in which a weight is only allowed to be non-positive;
- increasing - in which a weight is only allowed to be non-negative; and
- constant - in which a weight is only allowed to be zero.

Slope constraint is also used to restrict the possible values that a weight on a variable can take in a LTE. This constraint allows precise control of the partial derivative of the output variable with respect to the variable being constrained. Since the linear threshold approximant guarantees that the weight of the output variable is 1.0 (for monotonic increasing output variables) or -1.0 (for monotonic decreasing output variables), a bound on the partial derivative simply becomes a bound on the magnitude of the weight on the constrained variable.

Advantages of Atree 3.0 over backpropagation-type neural networks

Reduced number of calculations during training and evaluation

The "greater than or equal to" threshold on the linear expressions for both logic and linear threshold elements is called a *hard-limiter* and it replaces the sigmoid transfer function used in backpropagation neural network elements. The obvious advantage of using a hard-limiter is in speed of evaluation: a comparison operation is much faster than computing a sigmoid or even doing a table lookup to find a precalculated sigmoid value. The not so obvious advantage is that not all inputs of a logic gate have to be evaluated when using a hard-limiter. As soon as a zero input to an AND gate is found, we know the AND gate output is zero. As soon as a one input to an OR gate is found, we know that the OR gate output is one.

The effect of using hard-limiters is that the entire ALN does not need to be evaluated to determine the output of the tree. This contrasts with backpropagation-type neural networks, where every input of every layer must be evaluated before calculating the output.

Simplified adaptation during training

During the training, a logic gate employs a type of *heuristic responsibility* to determine which of its children is responsible for fitting a particular input point. This works its way down the logic tree until a single linear threshold element is made responsible for a data point. The linear threshold element adjusts its weights to reduce the square error of its fit to the point.

The fact that only one linear threshold element is made responsible for a data point is another difference between ALNs and backpropagation neural networks. The hard-limiters used in ALN elements make possible discrete choices of responsibility. In the backpropagation algorithm, the

use of sigmoids results in distribution of responsibility for error over all weights in the system (in theory at least).

An ALN limitations

In the two dimensional example illustrated in Figure B.14 with inputs x_1 and x_2, a trained ALN will output a logical one if an input pair (x_1,x_2) lies anywhere on or below curve, and a logical zero elsewhere. To find the value of the function $x_2 = f(x_1)$, we can fix the input x_1 to, say, 12.5 and try evaluating the ALN with successive x_2's until we find the x_2 that causes the ALN output to change from a logical one to a logical zero with a slight change of x_2. In the example, the value of x_2 that causes such a change is 16.2. In Atree 3.0, such an evaluation procedure is performed by a *binary search* which can work because the value of the ALN changes *only once* (from one to zero) as x_2 moves from $-\infty$ to $+\infty$. It can be said that the output of the ALN is *monotonic decreasing* with respect to x_2. The number of steps of binary search is determined by the precision required (derived from the number of quantization levels and the range of values of x_2.

In the example illustrated in Figure B.14, the inverse of $x_2 = f(x_1)$ is not a function, since there may be many different values of x_1 that correspond to a single value of x_2. Using the ALN relation approximant, we would find that the ALN output may change from zero to one and back again many times as we increase x_1 from $-\infty$ to $+\infty$ for a fixed value of x_2. In this case we can not use binary search to evaluate the inverse because Atree 3.0 does not have a method for evaluating inverses that are not functional, even though the relational approach has no difficulty with the representation of the inverse (one would just have to add parameters to binary search to start the search in a subinterval). In practical terms this limitation means that in the specification of the parameters for ALN training there should be at least one functional output which will be used for evaluation of the trained ALN tree.

INDUCTIVE LEARNING ALGORITHM (IL)[6]

The inductive learning program EMPIRIC is an implementation of the hierarchical mutual information classifier algorithm of Sethi and Sarvarayudu (1982). This algorithm forms a decision tree by maximizing the average mutual information gain at each partitioning step. This section will explain these concepts and their implementation.

In information theory, *information* is regarded as the removal of uncertainty. Thus, the occurrence of a likely event conveys less information than that of an unlikely event as there is less *a priori* uncertainty of its occurrence. If an alphabet X (all possible source outputs) consists of J symbols, the j-th symbol occurring with a probability of p_j, then the amount of information (in bits) associated with j is:

$$I_j = -\log_2 p_j$$

Shannon's (1948) entropy[7] is defined as:

---

[6]Most of the information in this section has been copied from the Ph.D. thesis by Ben Heller (Heller, 1992) since the program EMPIRIC, which implements inductive learning algorithm and which was developed by Ben Heller as a part of his thesis project was used in this study without any modifications.

$$H(X) = \sum_{j=1}^{J} p_j \log_2 \frac{1}{p_j} = -\sum_{j=1}^{J} p_j \log_2 p_j$$

Note: while it is generally accepted to write $H(X)$, H is not a function of x, but of the probability function p (Blahut, 1987).

In information theoretic terms, entropy is a measure of the average amount of information gained per symbol received in an alphabet, it can also be considered as a measure of the uncertainty as to the identity of a symbol before it is received. If all symbols are equally likely than the a priori (before the symbol is received) uncertainty as to the symbol is a maximum; the entropy being equal to the log of the number of symbols[8]. If one symbol becomes much more likely than the rest, then the uncertainty falls and the entropy asymptotically approaches zero.

Mutual information

If a communication channel has an output alphabet Y, with a probability $P_{j|k}$ that the j-th symbol of X was transmitted, given that the k-th symbol of Y was received, then the uncertainty as to j on receiving k is:

$$H_{j,k} = -\log_2 P_{j|k}$$

The *mutual information* is the reduction in uncertainty, i.e. the *a priori* uncertainty as to j less the *a posteriori* uncertainty upon receiving k:

$$I_{j,k} = -\log_2 P_j + \log_2 P_{j|k} = \log_2 \frac{P_{j|k}}{P_j}$$

The average mutual information is then the product of $I_{j,k}$ and the joint probability of (j,k) occurring, summed over all possible (j,k), i.e.:

$$I(X|Y) = \sum_{k=1}^{K} \sum_{j=1}^{J} p_{j,k} \log_2 \frac{P_{j|k}}{P_j}$$

The inductive learning problem is to choose a threshold t on an attribute a to discriminate between a set of examples Z with class values X. Y is obtained by applying the production rule:

$$'IF(a > t)THEN(y = 1)ELSE(y = 0)'$$

to an example z with class value x, i.e. the two possible values of Y are either 0 for attribute values below the threshold, or 1 for values above it. K is equal to 2, J is equal to the number of classes. $I(X|Y)$ is then the average mutual information gain for threshold t on attribute a at node N. By comparing this value with those obtained for all possible thresholds on all possible attributes at that node, it is possible to select the best discrimination rule (i.e. the rule which gives

---

[7]So-called because the function is the same as that used in statistical mechanics for the thermodynamic quantity entropy.

[8]When the logarithm is to base 2, the entropy is the maximum number of bits required to perfectly encode the alphabet.

the greatest gain in information/reduction in uncertainty) at that node. In the absence of true values for the probabilities, estimates are made based on the data in the training set.

To obtain the average mutual information for all the nodes in the decision tree T, the product of $I(X|Y)$ at a node and p_k, the probability of being at that node must be summed for all nodes:

$$I(X|Y) = \sum_{k=1}^{K} p_k I_k(X_k|Y_k)$$

Error rates

If the decision tree is to have a probability of classification error P_e, for J classes, then we can calculate $I_{\min}$, the minimum average mutual information required, as follows:

The average mutual information may be written:

$$I(X|Y) = H(X) - H(X|Y) \tag{B.3}$$

The maximum value of $H(X|Y)$, which occurs if all errors are equally unlikely, is given by inequality:

$$H(X|Y) \leq H(P_e) + P_e \log_2(J-1)$$

substituting this in (B.3):

$$I(X|Y) \geq H(X) - H(P_e) - P_e \log_2(J-1)$$

This represents the minimum gain in mutual information that will be provided by a decision tree having an error rate P_e. Thus, expanding for $H(X)$ and $H(P_e)$, becomes:

$$I_{\min} = -\sum_{j=1}^{J} p_j \log_2 p_j + P_e \log_2 P_e + (1-P_e)\log_2(1-P_e) - P_e \log_2(J-1)$$

Once the cumulative mutual information provided at each node exceeds this value, the decision tree has achieved the required error rate.

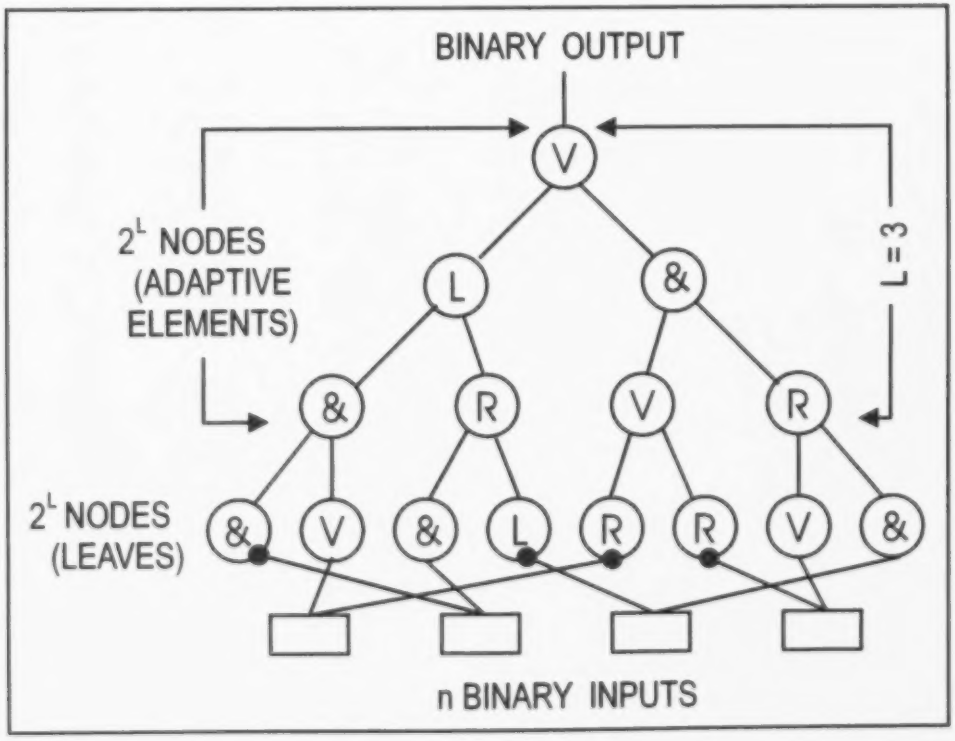

Figure B.1. Structure of binary tree representing idealized classification function

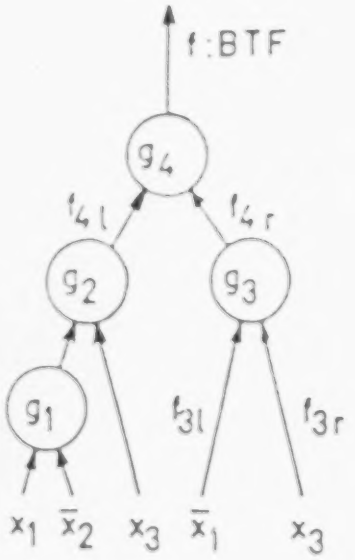

Figure B.2. Non-disjoint binary tree with inversions

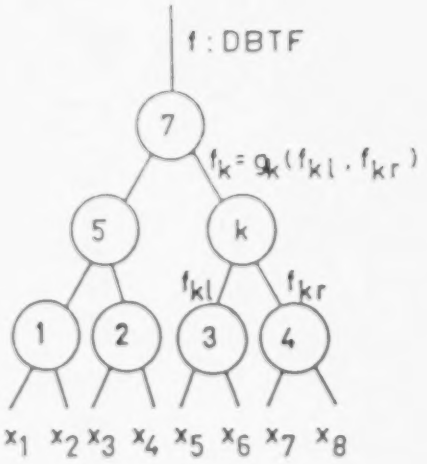

Figure B.3. Disjoint balanced binary tree

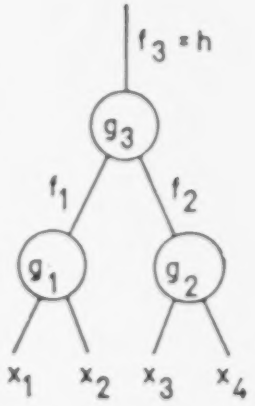

Figure B.4. An example of two-level tree

(from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong)

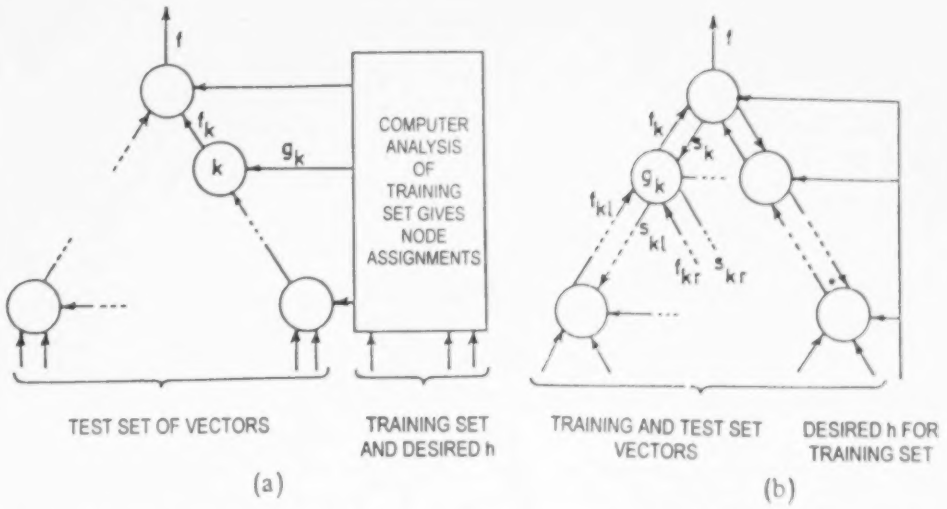

Figure B.5. (A) Computer-controlled generation of assignments. Node functions g_k are stores in memory and communicated to the tree. (B) System of adaptive elements. Each element receives two function values from its successors, a heuristic responsibility signal s_k from its predecessor and desired tree output value $h(\xi)$ (last two only during the training). Assignments g_k are generated in each node from the above signals.

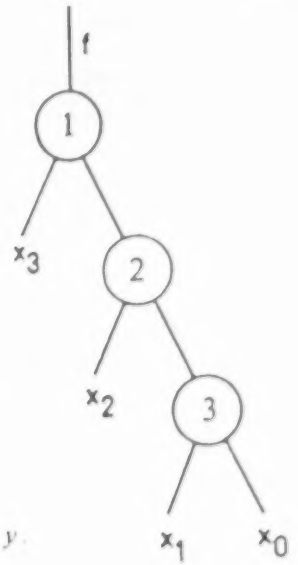

Figure B.6. Tree for threshold function $x \geq y$.

(from Armstrong and Gecsei, 1979, with permission obtained from W.W. Armstrong)

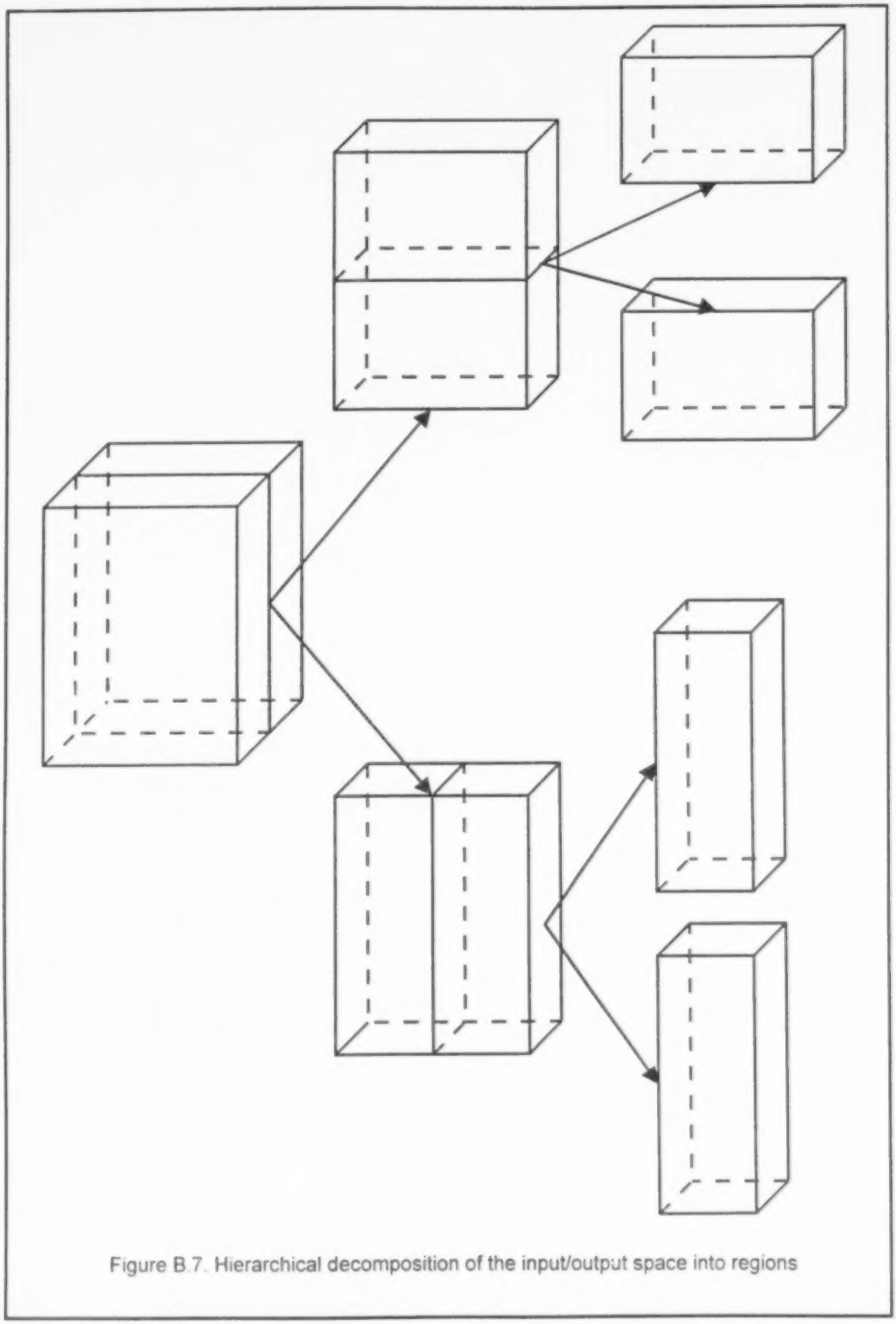

Figure B.7. Hierarchical decomposition of the input/output space into regions

ALN linear threshold element

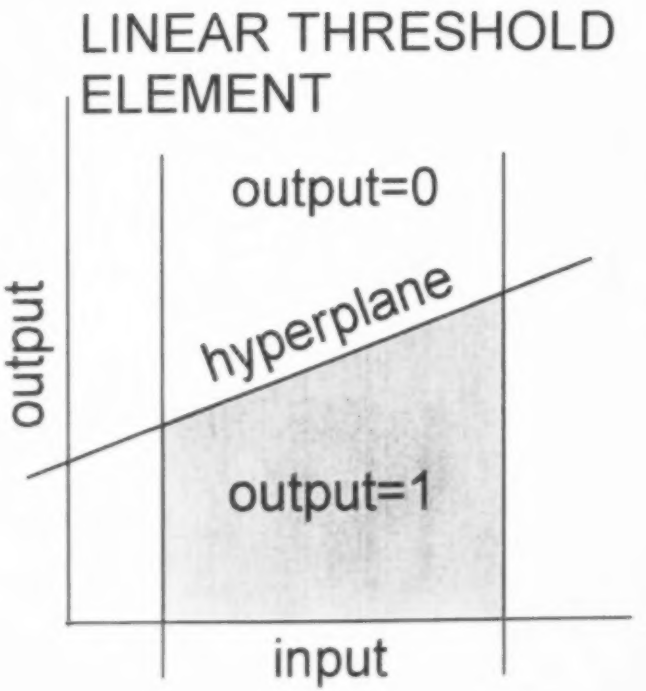

Figure B.8. Linear Threshold Element

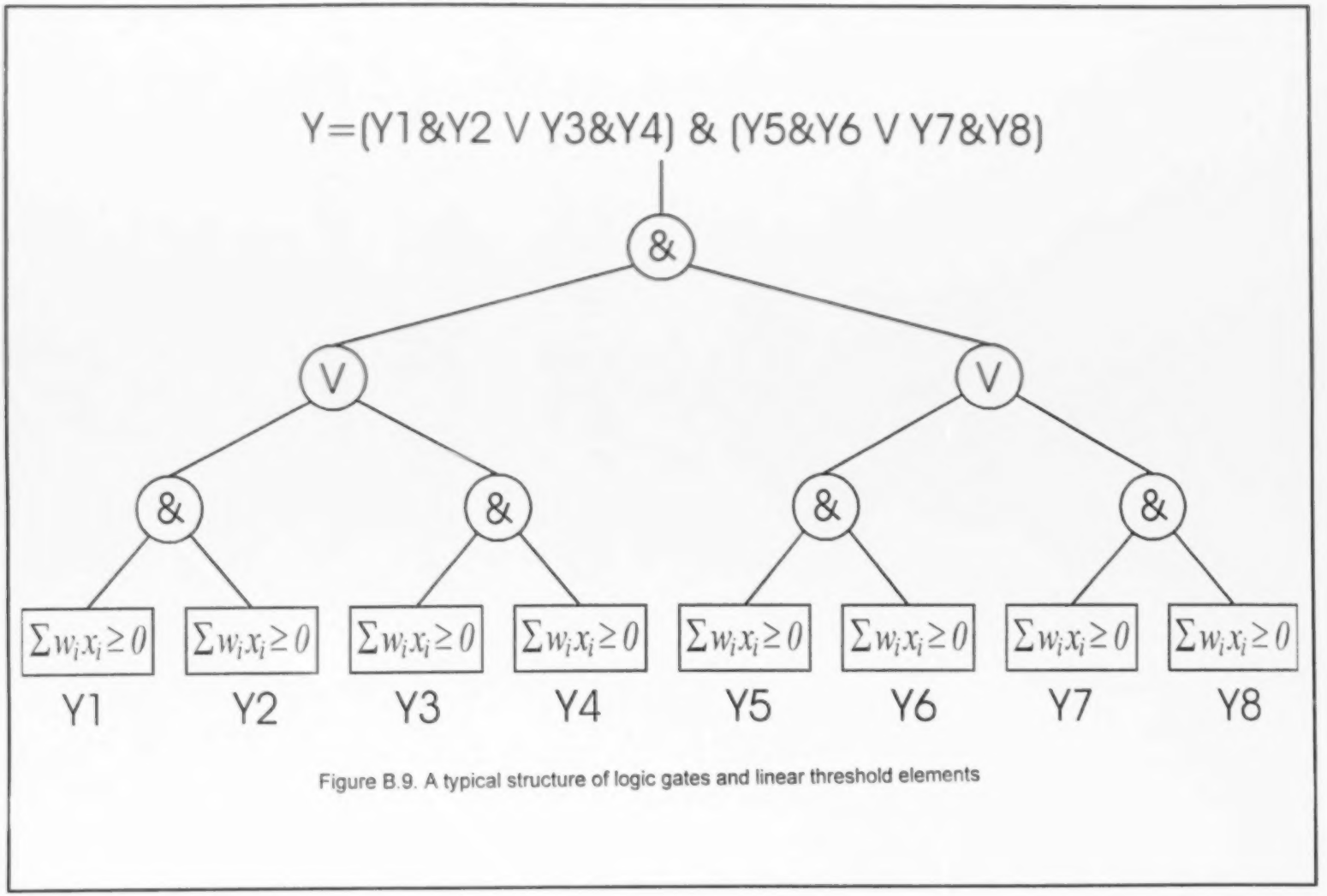

Figure B.9. A typical structure of logic gates and linear threshold elements

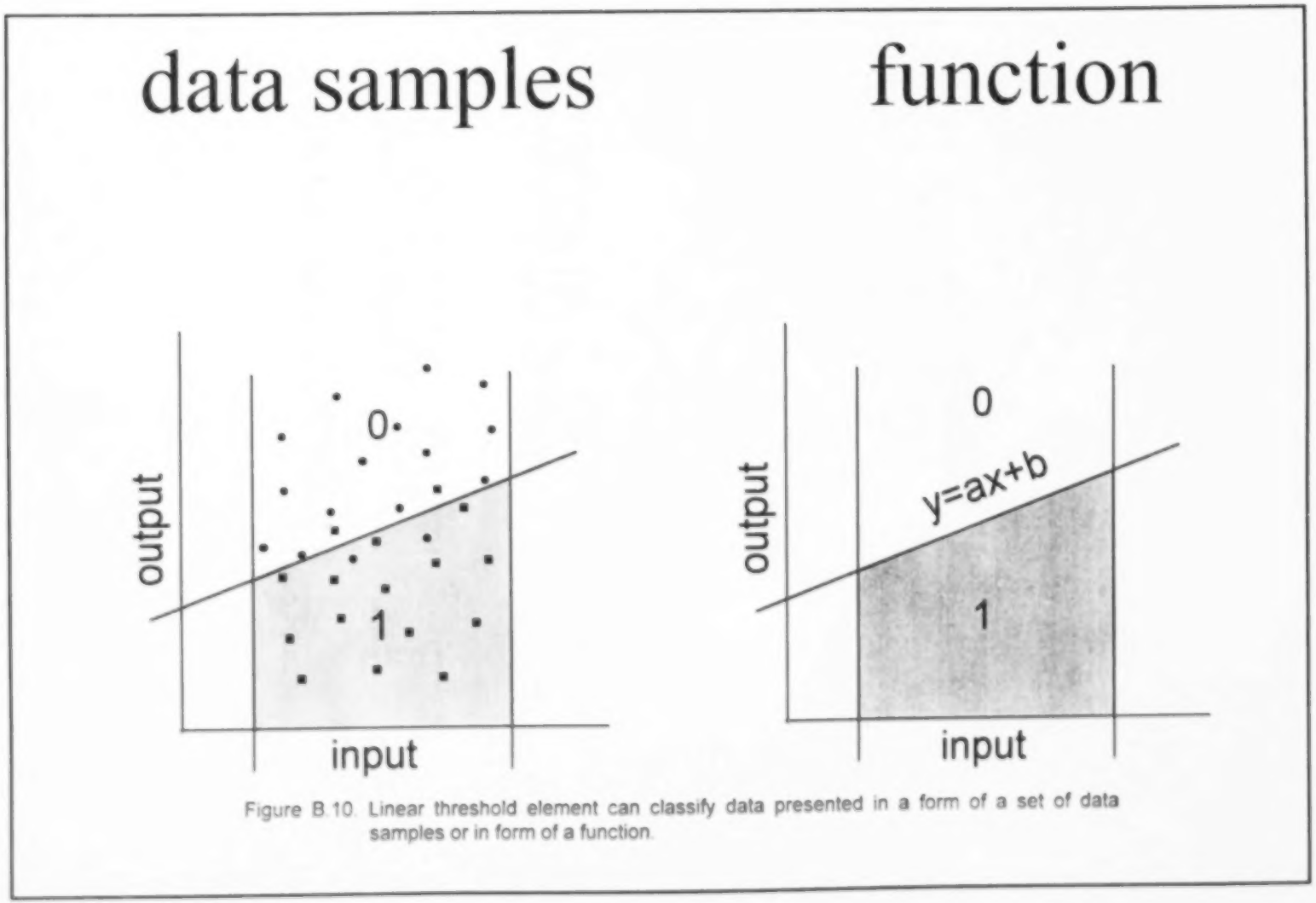

Figure B.10. Linear threshold element can classify data presented in a form of a set of data samples or in form of a function.

input/output relationship defined by a set of data samples

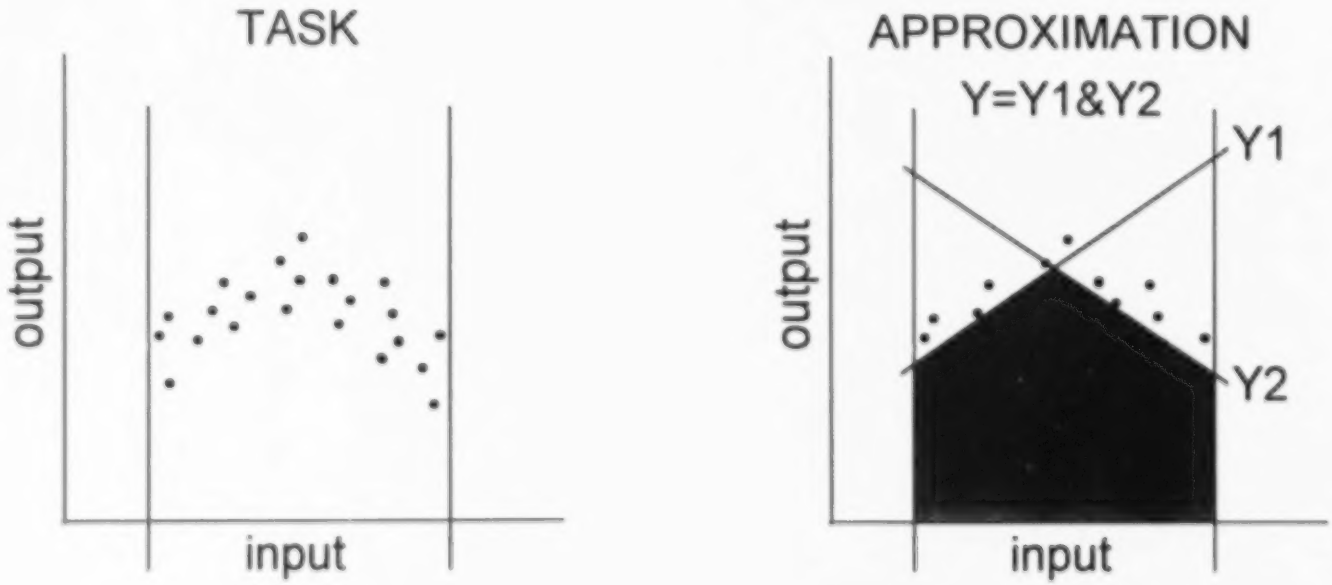

Figure B.11. Classification of data samples distributed in a non-linear way.

input/output relationship defined as a function

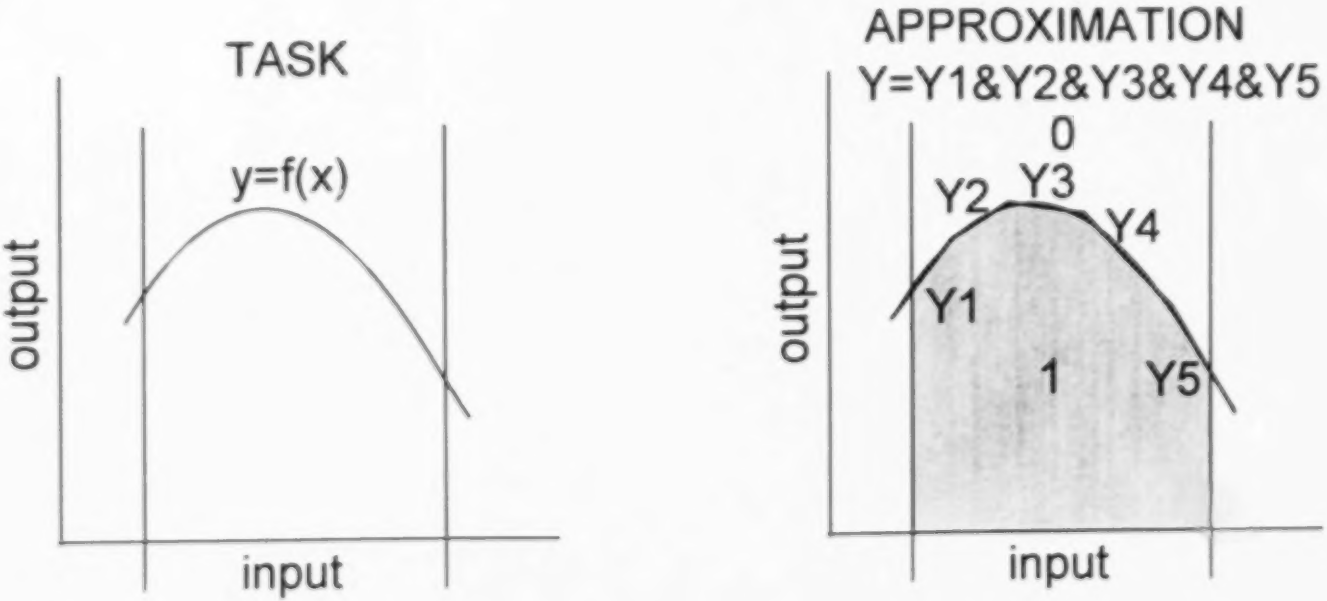

Figure B.12. An ALN approximation of convex-type non-linear functions

input/output relationship defined as a complex function

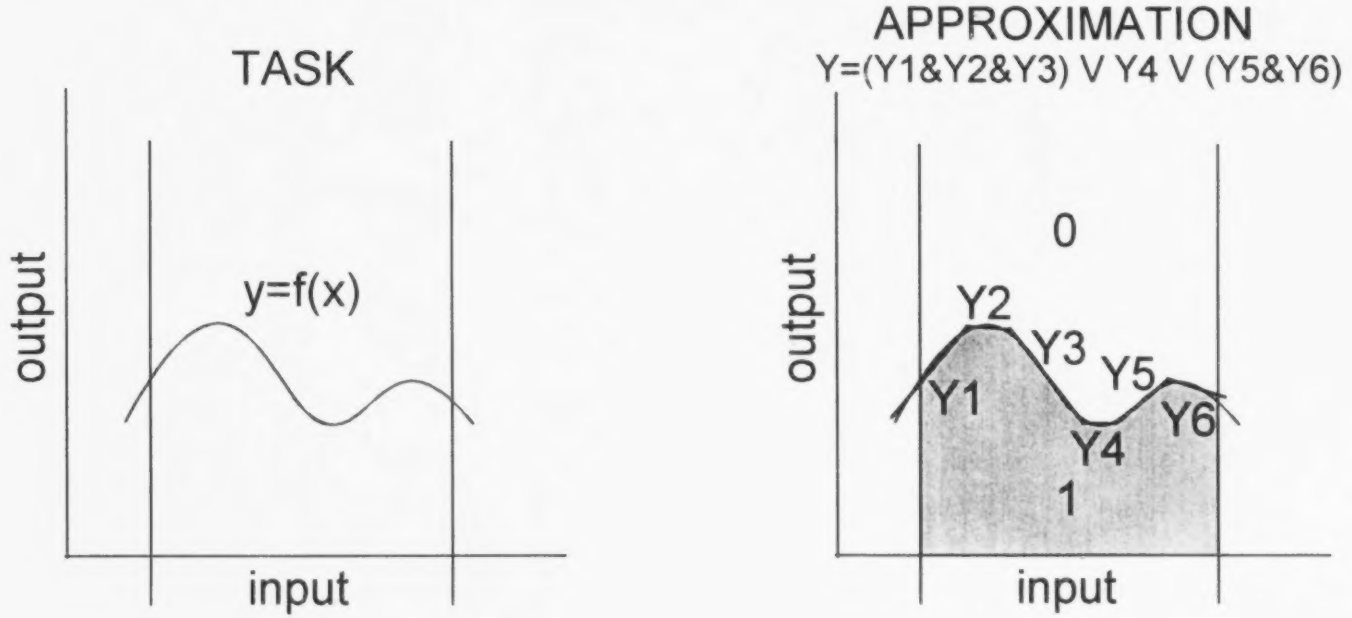

Figure B.13. An ALN approximation of a complex function.

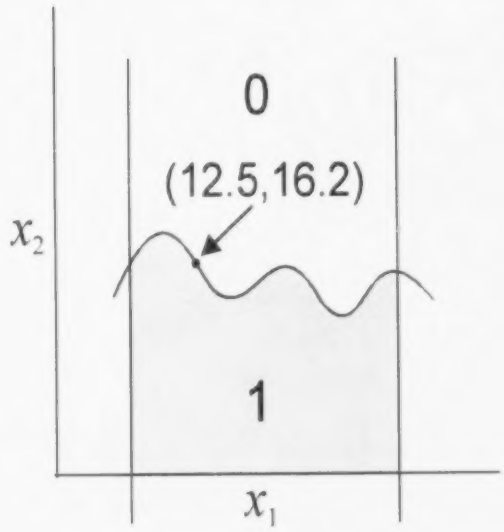

Figure B.14. An ALN approximation of a complex function which does not have inverse function.

Appendix C : PROGRAMS DEVELOPED FOR THIS THESIS PROJECT

HEX_FILT.EXE

This program is used to low-pass filter data stored in the Axotape FETCHEX 5.2 data files. It is written in programming language C. It does not change the data file format, which means that its output file is in the same FETCHEX 5.2 format and the filtered data still can be accessed by Axotape program. The program implements low-pass, zero-phase shift, dual-pass, fourth-order Butterworth digital filter (Barr and Chan, 1986) to selected data channels stored in a file.

To use the program, the DOS command in the following syntax should be issued:

hex_filt *Original_Axotape_file_name Filtered_Axotape_file_name <Enter>*

where:

hex_filt is the name of the executable form of the program which should include the full path if the program is not in the same directory as the data file and if the program's path is not listed in the PATH command of the computer's AUTOEXEC.BAT file;

Original_Axotape_file_name is the name of the Axotape FETCHEX data file which should include the full path if the data file is not in the same directory where the command was issued; and

Filtered_Axotape_file_name is the name of the new filtered data file which should include the full path if the new data file was to be created in a different directory from the one in which the command was issued.

HEX_ASC.EXE

This program reads the specified Axotape FETCHEX 5.2 data file and converts it into ASCII data file of the following WALKON format:

ChannelFile *N*

!*Channel_1_Name*! μ

!*Channel_2_Name*! μ

...

!*Channel_N_Name*! μ

X_{11} X_{12} ... X_{1N}

X_{21} X_{22} ... X_{2N}

...

X_{M1} X_{M2} ... X_{MN}

END

where:

N is the total number of channels in the data file;

Channel_N_Name is the name of the *N*th channel;

μ is the code for particular channel type and it can take the following values: 0 - for "Not Used", 1 - for "Domain", and 2 - for "Codomain";

X_{ij} is the *i*th data sample of the *j*th channel;

M is the number of samples in the file;

ChannelFile and **END** are words used to mark explicitly the file type and the end of the file respectively;

exclamation mark (!) is used to delimit the data field for the name of the channel and it is the only character that can not be used as a part of the name of the channels;

all information in the same line of the header is separated by single space character and the data columns (data channels) are separated by single Tab character.

To use the program, the DOS command in the following syntax should be issued:

hex_asc *Axotape_file_name ASCII_file_name <Enter>*

where:

hex_asc is the name of the executable form of the program which should include the full path if the program is not in the same directory as the data file and if the program's path is not listed in the PATH command of the computer's AUTOEXEC.BAT file;

Axotape_file_name is the name of the Axotape FETCHEX data file which should include the full path if the data file is not in the same directory where the command was issued; and

ASCII_file_name is the name of the ASCII data file which should include the full path if the data file was to be created in a different directory from the one in which the command was issued.

In addition to its primary function, the program can be also used to decimate the Axotape data file. After reading the header of the Axotape data file and presenting it on the computer screen, the program asks for the reduction factor, which should be a non-zero integer. No decimation (reduction factor equals to one) is assigned by default, which means that if decimation is not required, it is enough just to press on Enter key to proceed.

PROGRAM "WALKON"

Program's Concept

Program WALKON contains all functions that are not grayed in Figure 2.8.1. Data acquisition and storage functions are done in advance using Axotape system (Axon Instruments, Inc) and real-time control was not implemented in this program. It can retrieve and save ASCII files in specific format, which will be called for simplicity "Walkon" format and which is described in previous section. The program can also import data from ASCII files prepared in matrix form, where the signal channels are organized in columns separated by Tab character, and samples in rows.

functions can be activated by selecting an appropriate icon on the customized toolbar. Since the program uses extensively MS Windows GUI (Graphics User Interface), the primary human interface to the program consists of the pointing device (computer mouse) and the computer screen. Computer keyboard is mainly used to enter or edit character-based information. The program has an extensive context-sensitive on-line help which can be used mainly as a reference. Program tutorial is planned, but it is not yet realized.

Program Requirements

Computer: Any 386, 486, PS/2 or Pentium computer that will run Microsoft Windows 3.1 or higher in Enhanced Mode. At least 4 MB RAM is required, 16 MB suggested if processing large data sets and training large ALNs.

Software: an installed copy of Microsoft Windows, version 3.1 or higher.

Display: any Windows-supported video system of VGA or better resolution.

Mouse: any pointing device supported by Windows. Although the program can be operated entirely from the keyboard, three-button mouse with double-click emulation on a middle button is recommended for optimum convenience and speed.

Windows Conventions

- A click means a single press of the left mouse button.
- A double click means two clicks of the left mouse button in rapid succession.
- Dragging the mouse means clicking and holding the left button while moving the mouse in the desired direction.
- The active window is the window having a highlighted border and title bar.

Program Installation

The program is stored in subdirectory WALKON on the diskette contained in a pocket on the front cover of this thesis. To install the program, start Windows, start File Manager, select the drive in which is the installation diskette, double click on the folder representing subdirectory WALKON, locate the install program INSTALL.EXE and double click on its name. Install will prompt you for a home directory for WALKON 2.1. It will uncompress the distribution files to that directory, and will create a Program Manager group for WALKON 2.1.

Running the Program

To start the program, double click on the WALKON icon in corresponding program group of the Program Manager. Opening screen will show up, such as one illustrated in Figure C.1.

The program has the following menu groups and items:

File

New file
Open file
Save file
Save file as
Preferences
Import V.1 file
Import text file
Export text file
Exit

Edit
- Cut
- Copy
- Paste
- Delete

Channel
- File Encode
- File History Points
- Stack
- Compare
- Differentiate
- Shift
- Truncate
- Threshold

ALN
- New
- Edit Parameters
- Train
- Evaluate

Window
- Cascade
- Tile
- Arrange icons
- Close all
- (titles of currently open windows)

Help
- Index
- Using help

About

Customized toolbar has icons which are linked to the following functions:

- About: Display version and copyright information;
- Open: Locate and open channel or tree files;
- Save: Save current channel file;
- Cut: Remove a channel from current file and put it into clipboard;
- Copy: Copy a channel from current file and put it into clipboard;
- Paste: Paste a channel from clipboard into current file;
- Stack: Display all the channels from the current file;
- Train: Train or retrain an ALN;
- Evaluate: Evaluate current ALN;
- Help: Display an index of on-line help topics;
- Exit: Exit the program.

Data manipulation functions

Data retrieval can be done either by using *Open File* and *Import data* functions located in menu group *"File"* or by single click on the *Open* icon on the toolbar. These functions are standard Windows functions which are described in every MS Windows user's manual. After the data are retrieved, new data window opens and the functions and parameters of the signal preprocessing,

quantization and encoding become available in form of customized spreadsheet (see Figure C.2). Signals are called channels and they are organized in rows of the spreadsheet. Their characteristics and functions that can be performed on, or with a single channel are organized in columns. The first column is an icon used to display single channel. Next field is channel name, which can be edited. The third column contains a button entitled *Encode* which is used for single channel encoding. The fourth column is a pull-down list for selection of channel type in ALN training, and the fifth column contains button used to define history points (past samples) in a single channel. Those functions that can be performed on multiple channels are grouped under menu title *"Channel"*.

Signals can be diagramatically displayed in separate windows or stacked together in single window. To display a single channel, click on the first icon in the appropriate channel row of the spreadsheet. This function will open new *Graph* window as the one shown in Figure C.3. This form of graphics display is very useful to set relevant history points, which can be displayed in different color or hidden.

To display all of the channels, click on the icon *Stack*. The active window will be maximized and all channels will be displayed stacked one above another. An example of five signals stacked is shown in Figure C.4. This window allows for displaying history points and truncating the file. It also displays the name of the channel and the current coordinates of the mouse cursor at every instant, which can be used to decide what to truncate from the current file and where.

Previous samples in single channel can be defined using function *Set Hpoints* or in all channels using menu function *File History Points*. A dialog box will open with options to add new or delete existing history points (see Figure C.5).

As mentioned above, those history points already set can be viewed, either in single channel display or using *Stack* function (see Figure C.6).

Parts of the data set can be cut out using function *Truncate Channels*. Truncation is reflected to all channels in a data file due to the restriction that all data samples have same channels. Truncating parts of the signal that are of no interest for particular ALN training session can be done either from within *Stack* window (see Figure C.7) or using the function *Truncate* from menu group *Channels*. History points are produced by shifting original signals to the right, so that some of the samples at the beginning and at the end of the file are filled with zeros. These samples should be cut out from the data file before the training. Truncation is also required when shifting domain signals to the right for future events prediction.

A type of encoding and related parameters can be specified using function *Encode*. Two types of encoding are available: *Random Walk* and *Unary* (called *Linear* in the program). After the type of encoding is chosen, various encoding parameters can be applied to different channels or all channels can be encoded the same way using *File Encode* function located in menu group *"Channel"*. Figure C.8 illustrates ***Random Walk*** type of encoding applied to a single channel. Parameters that have to be specified are: *Minimum*, *Maximum*, number of *Quantization levels*, binary vector *Width*, and a *Step Size*. Graph in an Encode dialog box automatically displays the signal and corresponding quantization levels.

The second type of encoding available in the program is ***Unary (Linear)*** encoding. It is illustrated in Figure C.9. The parameters for this type of encoding are displayed in the same dialog box, described for Random Walk encoding. In this type of encoding the number of Quantization levels is the most important parameter because it also defines the width of the binary vector. Single channel encoding dialog box also provides display of the signal and quantization levels.

In addition to uniform distribution of the quantization levels, *Unary (Linear)* encoding allows for arbitrary positioning of quantization levels (see Figure C.10). Levels can be specified numerically or by using a pointing device (computer mouse). This feature enables a researcher to specify uneven distribution of quantization levels, which can be used to exploit more efficiently same number of quantization levels.

To produce various delays between *Domain* and *Codomain*, which is useful for prediction of future events, there is a function *Shift* located in the menu group *"Channel"*. Another function, called *Differentiate* replaces the original signal in the selected channel with differentiated signal. These two functions have similar user's interface, illustrated in Figure C.11. To produce differentiated channel and at the same time to keep the original one, is possible by reproducing (*Copy* and *Paste*) the original channel before one of the copies is differentiated.

Standard MS Windows functions grouped under menu *Edit* apply to channels. They enable the user to *Cut*, *Copy* and *Paste* channels, one at the time. The same functions are available as icons in the toolbar.

ALN-related functions

To define input data set for ALN training, each channel should be characterized either as *Domain*, *Codomain* or *Not Included*. This function is available in *Channel Type* column of the data file spreadsheet.

After the data set is prepared for training, a *New* ALN can be selected from *"ALN"* menu. Training parameters are available in two dialog windows. In the first one the number of presentations of the data set to the learning algorithm, called *"# Epoch"* and the allowed training error, called *"% Correct"*, are selected (see Figure C.12). The same dialog window lists ALN tree parameters, which can be accessed by clicking on the button marked *Tree/Func Param's*.

The second dialog window gives access to the ALN tree parameters (see Figure C.13). Here, the user specifies *"# Leaves"*, which defines the size of each binary ALN tree, and *"# Voters"*, which defines the number of parallel trees that will be trained to produce the same *Codomain* bit. Both windows list all *Domain* and *Codomain* channels and the second one still allows for changes of channel status (*Domain, Codomain, Not Included*).

After all training parameters are decided, the actual ALN training can be started. During the training, a message window reports on the current training *Epoch* number and the training progress. As soon as the number of correctly learned bits reaches the preset training error, the training is stopped and the total training time is displayed in the window (see Figure C.14). This window can be displayed always on top of the other open windows, which may be useful for monitoring the ALN training while working in some other program in Windows pseudo multi-tasking environment.

In addition to training new ALNs, the program can be used to retrain existing ALNs with the same or new data set. Trained ALN trees are saved in binary format and they can be retrieved and used for evaluation, or retrained.

To assess the performance of the trained ALNs on the training set, they can be evaluated using the function *Evaluate* shown in Figure C.14. This function produces new data set and new spreadsheet which consists of only one channel automatically named *"Result: actual_channel_name"*. A restored signal from *Codomain*, produced by ALNs is placed in that channel (see Figure C.15).

To compare original set of values from the *Codomain* and the one produced by ALN, there is a function *Compare* located in menu group *"Channels"*. This function can operate only within one data file. Since *Result* and original *Codomain* are not in the same data set, either one should be copied from its data set (and spreadsheet) to another one. Function *Compare* produces new channel with a signal representing the difference between two signals compared (see Figure C.16).

The error of ALN prediction can be measured using the function *Threshold* located in menu group *"Channel"*. This function operates on a single channel and counts all samples whose amplitude is higher than high threshold and lower than low threshold (see Figure C.17). High and low

thresholds can be set arbitrarily either by typing the new position in the corresponding field or by pointing to a new position on a graphics display with a pointing device (computer mouse).

To test the performance of the trained ALNs on a new data, test set can be loaded and ALNs evaluated in the same way.

PROGRAM "FESCONT"

The program is still in development phase (functional prototype) and it will not be presented in details. It incorporates and functionally integrates all functions presented in Figure 2.7.1. To provide continuity with programs used during the MLTs evaluation phase, function calls to the ALN-related functions, stored in form of Windows DLL (dynamic-link library), are enclosed. ALN functions listed are: LVALNCreateALN, LVALNDestroyALN, LVALNTrainALN, LVALNExecALN, LVALNSaveALN, and LVALNLoadALN. The names of these functions are self-descriptive enough. Parameters used in this function calls are as follows:

nLayers	number of layers in ALN trees
nVoters	number of ALN voters
nDim	number of dimensions, total number of domain and codomain channels
nFanin	fanin of the ordinary nodes in ALN trees
nLastFanin	fanin of the top node in ALN tree
bOR	top node is OR (true) or AND (not true)
adblMax	an array of maximum amplitudes in each channel
adblMin	an array of minimum amplitudes in each channel
nOutputIndex	index of the codomain (zero based)
hALN	an ALN handle
nEpochs	number of epochs
dblMinPctCorrect	minimum percentage of correct predictions during ALN training
pdblData	two dimensional array of actual data, channels in columns
nSamples	total number of samples
dblLearnRate	learning rate (described in ALN V.3 Section of Appendix B)
wQuant	number of quantization levels for the ALN output
szFileName	file name of the ALN tree

The actual header file of the LabVIEW - ALN interface DLL (LVALN.DLL) is as follows:

```
// lvaln.h
// Copyright © 1993 Dendronic Decisions Limited

#ifndef __LVALN_H__
#define __LVALN_H__

#ifdef __cplusplus
extern "C"
{
#endif
```

```
// variable creation

// aln creation
// LabVIEW signature: WWWWWW11W - 32 bit return
DWORD WINAPI LVALNCreateALN(int nLayers, int nVoters, int nDim, int nFanin, int nLastFanin,
int bOR, double far* adblMax, double far* adblMin, int nOutputIndex);

// aln destruction
// LabVIEW signature: I - 32 bit return
DWORD WINAPI LVALNDestroyALN(DWORD hALN);

// aln training
// LabVIEW signature: IWD2WD - 32 bit return
DWORD WINAPI LVALNTrainALN(DWORD hALN, WORD nEpochs, double dblMinPctCorrect,
double far* pdblData, WORD nSamples, double dblLearnRate);

// aln execution (single row of data)
// LabVIEW signature: I1W - 32 bit return should be cast to float
DWORD WINAPI LVALNExecALN(DWORD hALN, double far* pdblData, WORD wQuant);

// aln save
// LabVIEW signature: IS - 32 bit return
DWORD WINAPI LVALNSaveALN(DWORD hALN, LPCSTR szFileName);

// aln load
// LabVIEW signature: S - 32 bit return
DWORD WINAPI LVALNLoadALN(LPCSTR szFileName);

#ifdef __cplusplus
}
#endif

#endif __LVALN_H__
```

Figure C.1. Opening screen of the program WALKON.

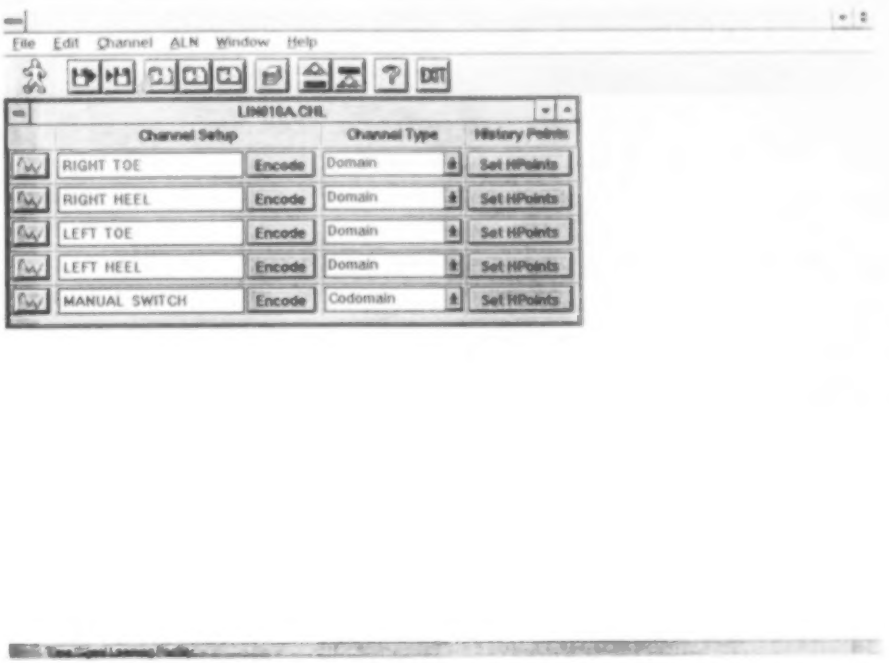

Figure C.2. Opening a new data file.

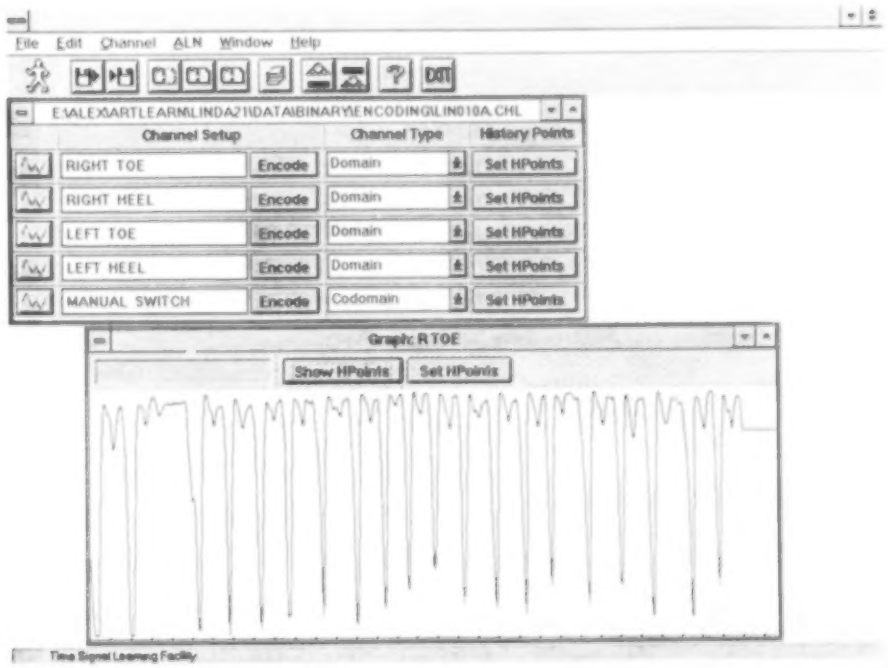

Figure C.3. Display single channel.

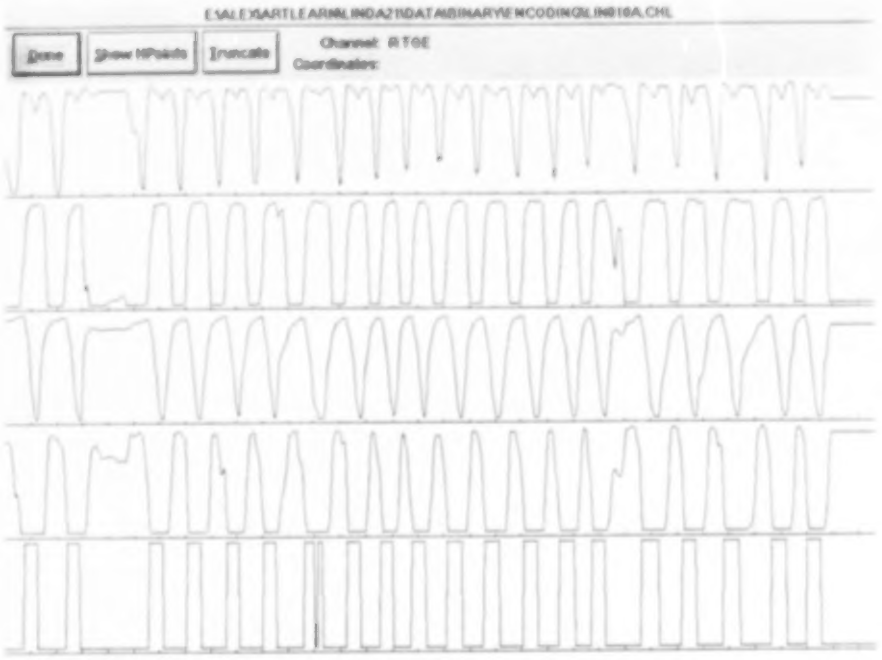

Figure C.4. Stack all channels.

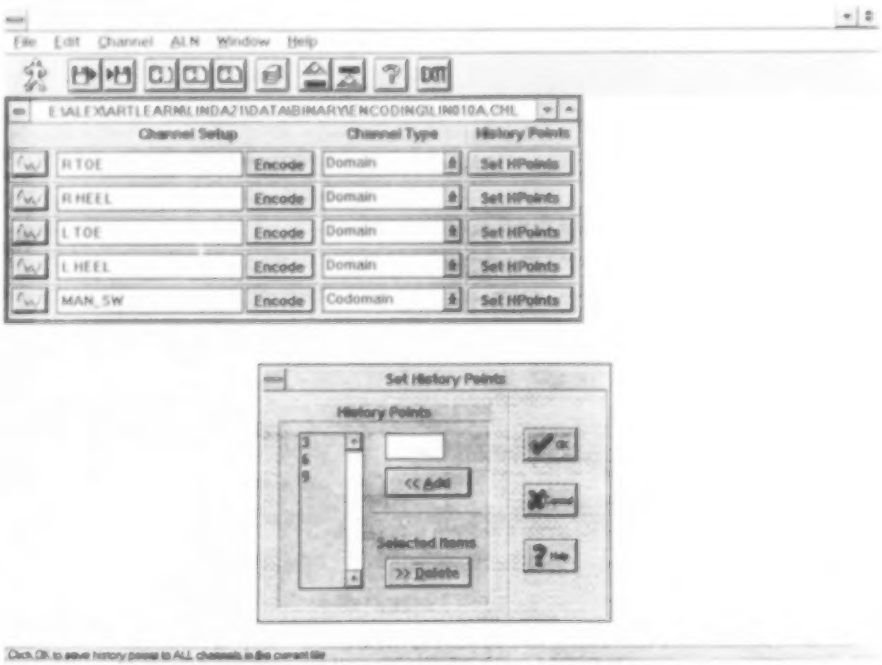

Figure C.5. Set history points dialog box.

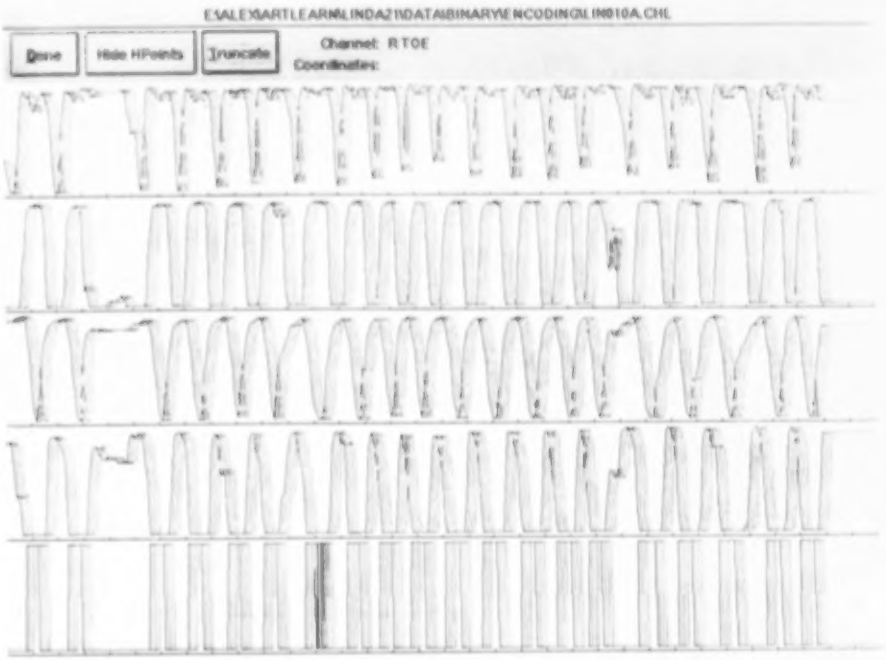

Figure C.6. History points display using *Stack* function.

Figure C.7. Truncate channels dialog box.

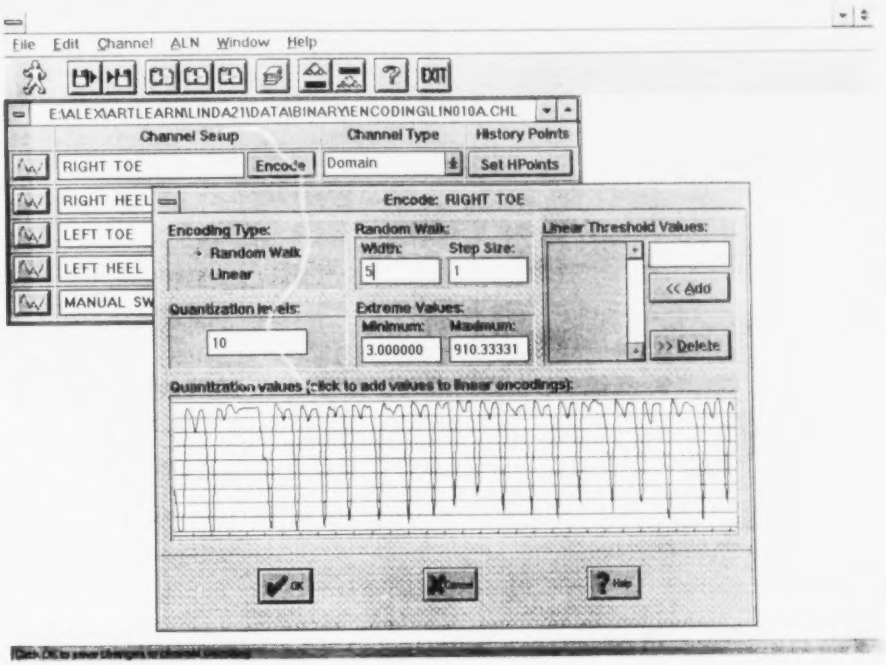

Figure C.8. Random Walk encoding.

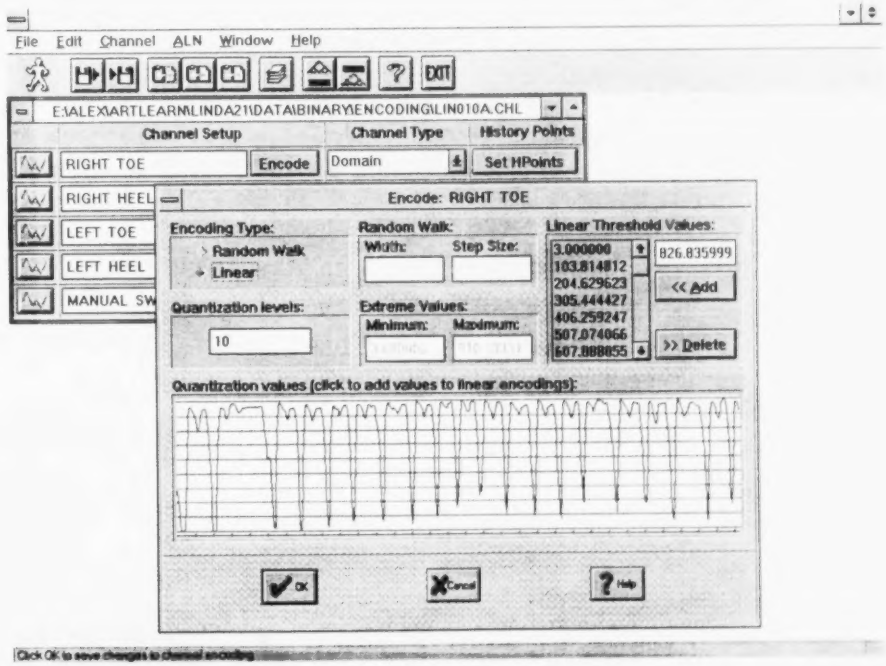

Figure C.9. Unary (linear) encoding.

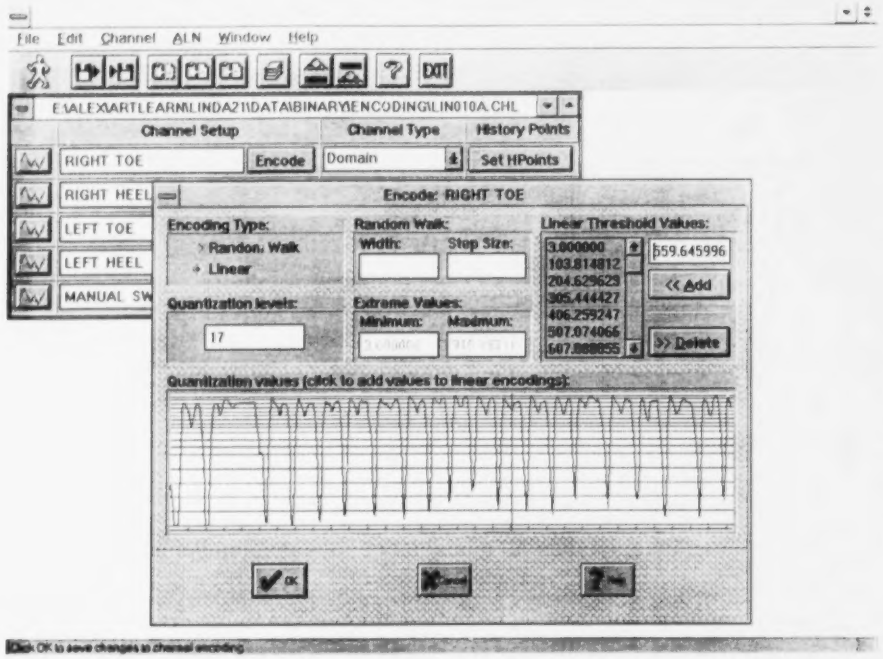

Figure C.10. Arbitrary position of quantization levels over the amplitude range.

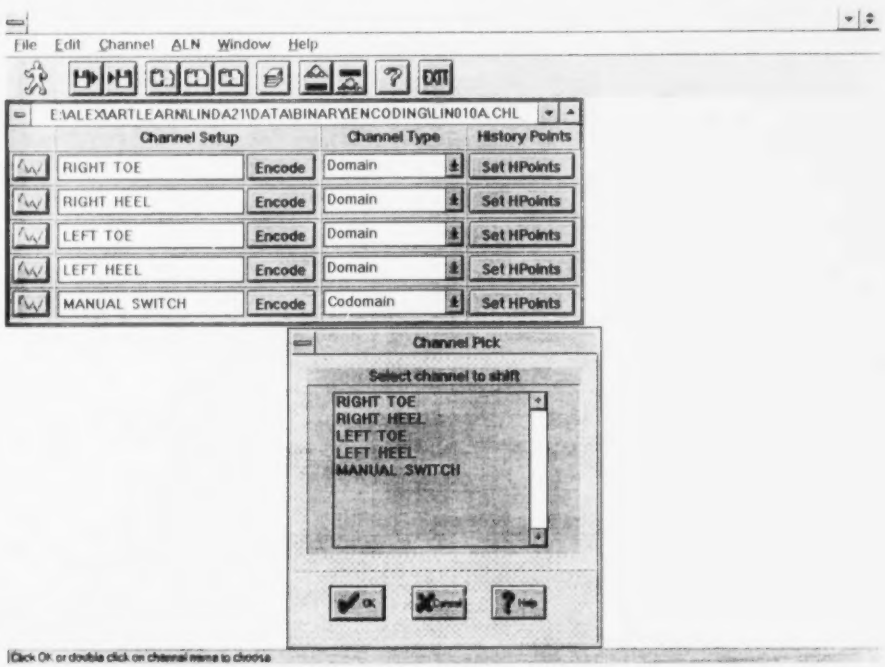

Figure C.11. Dialog box for choosing a channel to shift. Similar dialog box is used for choosing a channel to differentiate.

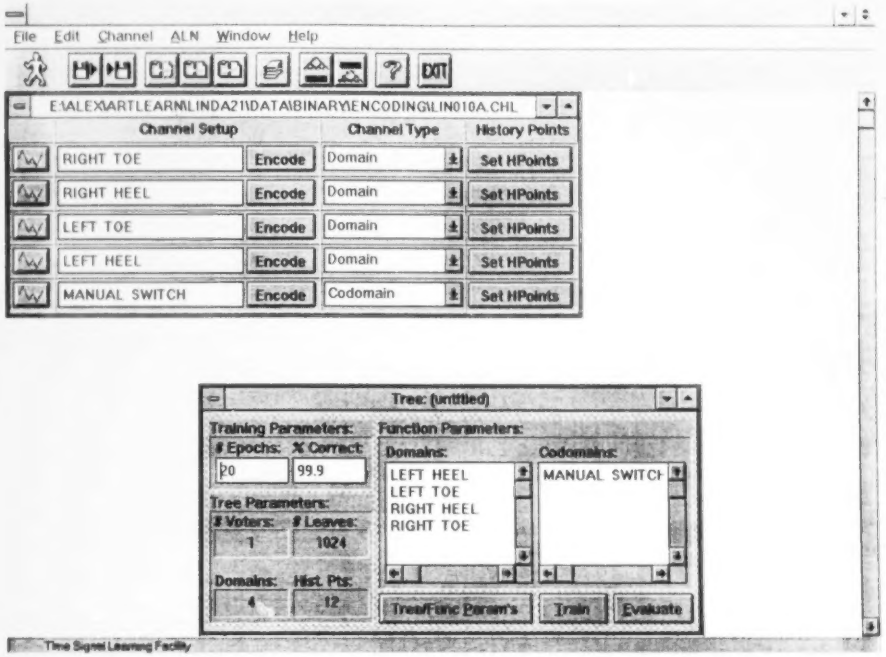

Figure C.12. Dialog window for specification of ALN training parameters.

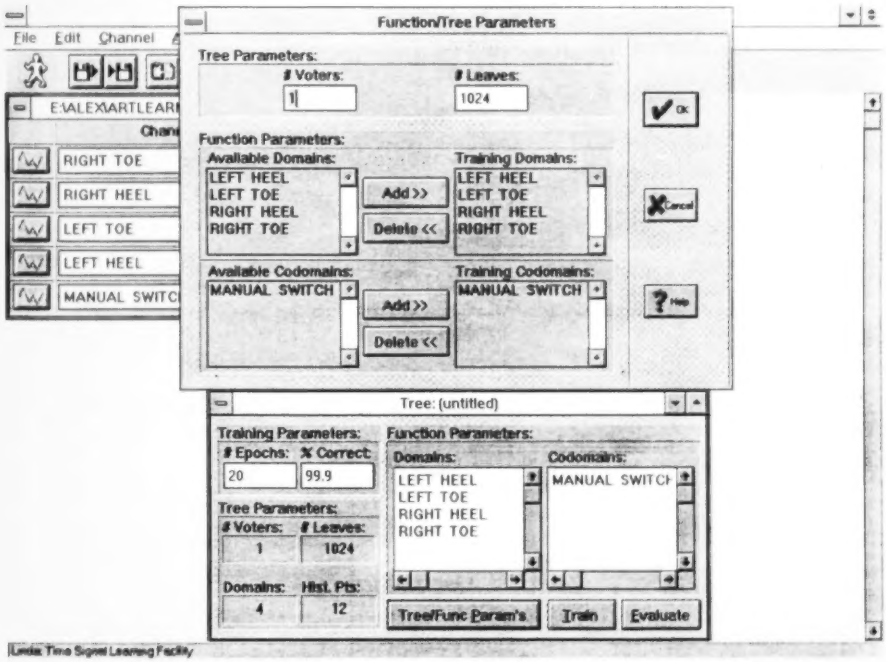

Figure C.13. Dialog window for specification of ALN tree parameters.

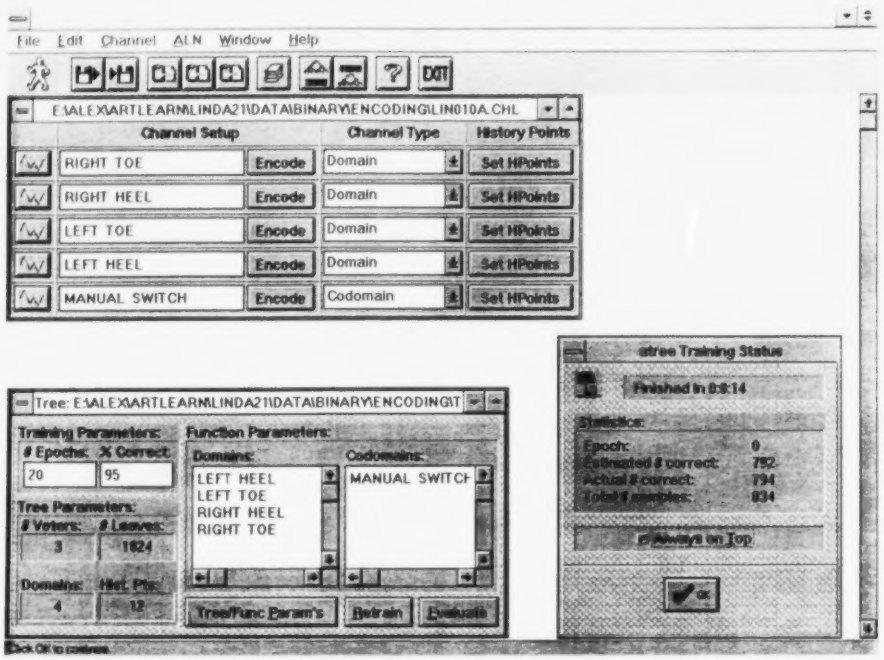

Figure C.14. ALN training status window after the training was finished.

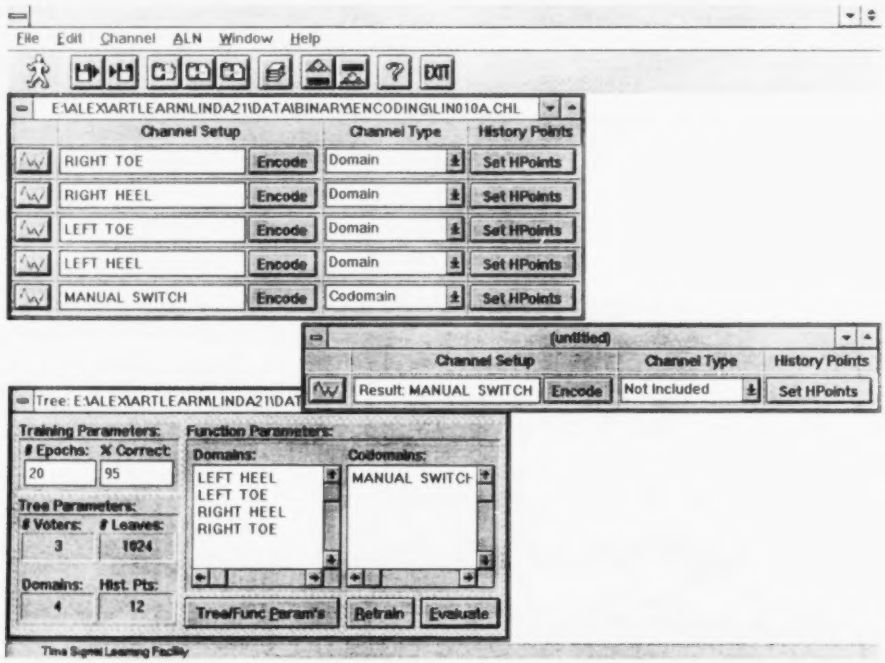

Figure C.15. Evaluation of the trained ALN trees results in new data file spreadsheet (untitled).

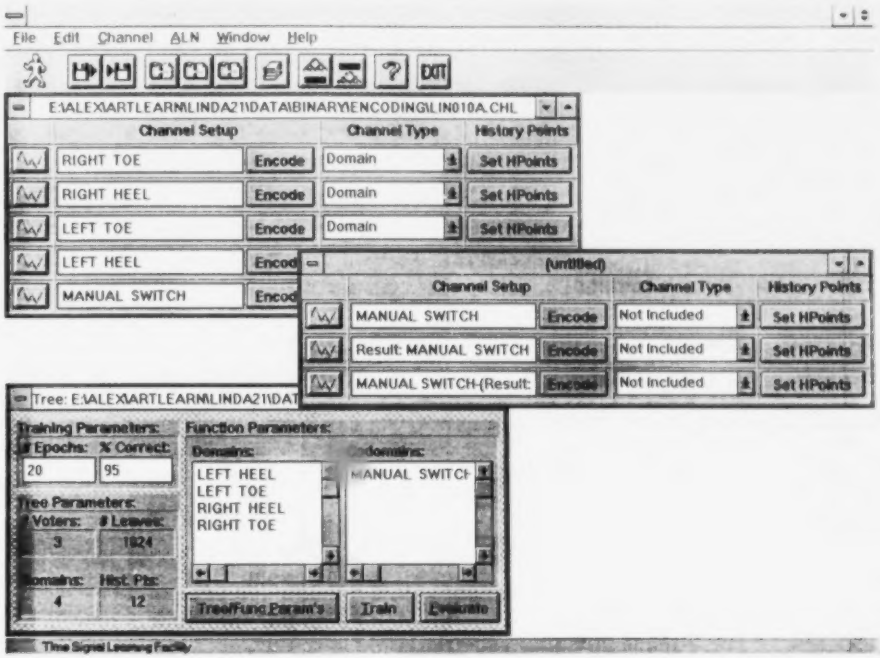

Figure C.16. Original stimulation control signal, the signal predicted by ALN and signal of their difference are organized in a new data file.

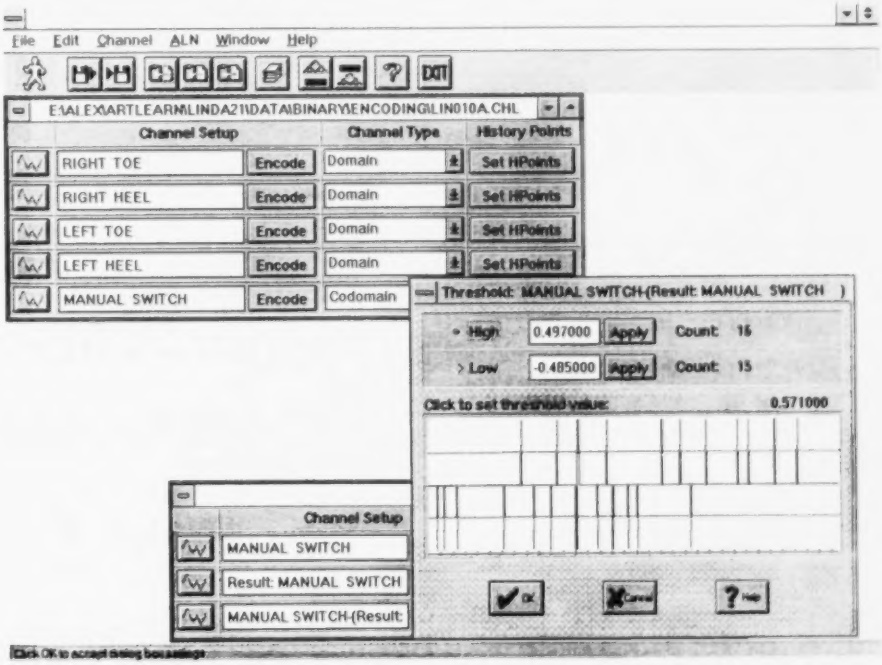

Figure C.17. Assessing the performance of trained ALNs.